OIPE IAP17 MAY 23 2007 PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	)	
Richard Francis Russell, et al.	)	Group: 2142
Serial No.: 09/957,014	)	
Filed: September 20, 2001	)	
Title: AUTOMATIC REMOTE ASSIGNMENT OF INTERNET	)	
PROTOCOL ADDRESS INFORMATION TO A NETWORK	)	Examiner: B. Prieto
DEVICE	)	

SECOND SUBMISSION OF REPLACEMENT BRIEF OF APPELLANT

MS APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This appeal is taken from the decision of the Examiner, dated October 4, 2005, finally rejecting claims 1-25, all of the claims that are under consideration in the above-captioned patent application. Appellants timely filed a Notice of Appeal in this matter on January 3, 2006. Appellants received a Notification of Non-Compliant Appeal Brief (37 CFR 41.37), mailed December 1, 2006, which asserts that Appellants' Brief does not comply with 37 CFR 41.37(c)(1)(v). Appellants respectfully submit that their Brief of Appellant, as originally filed, complies with 37 CFR 41.37(c)(1)(v). Nonetheless, Appellants revised the form of their Brief to explicitly identify by number the claims summarized in the "Summary of Claimed Subject Matter" section, re-arranged the references to their specification and drawings to provide additional clarification, and submitted a Replacement Brief on January 3, 2007. Appellants presently resubmit their Replacement Brief of Appellant in response to a second Notification of Non-Compliant Appeal Brief (37 CFR 41.37), mailed April 23, 2007, which Appellants contest for the reasons set forth in the Letter submitted herewith.

I. TABLE OF CONTENTS

Real Party In Interest ..... Page 3

Related Appeals and Interferences ..... Page 4

Status of Claims ..... Page 5

Status of Amendments ..... Page 6

Summary of Claimed Subject Matter ..... Page 7

Grounds of Rejection To Be Reviewed On Appeal ..... Page 11

Argument ..... Page 12

Claims Appendix ..... Page 47

Evidence Appendix ..... Page 52

Related Proceedings Appendix ..... Page 53

II. REAL PARTY IN INTEREST

The real party in interest in this appeal is Lexmark International, Inc., a corporation organized and existing under the laws of the State of Delaware, which owns the entire interest in this patent application as set forth in the underlying claimed invention.

III. RELATED APPEALS AND INTERFERENCES

No related Appeals or Interferences are known to the Appellants.

IV. STATUS OF CLAIMS

Pending: 1-25.

Canceled: None

Allowed: None.

Objected To: None.

Rejected: 1-25.

Withdrawn from Consideration: None.

On Appeal: 1-25.

V. STATUS OF AMENDMENTS

A Reply Under 37 CFR 1.116 was submitted in this case on August 17, 2005, in response to the final rejection in the Office Action mailed June 17, 2005. The Reply did not include any claim amendments. The Reply was not entered, as indicated in the Advisory Action mailed September 14, 2005 and as discussed in an Interview with the Examiner on September 12, 2005. A Request For Continued Examination was filed on September 13, 2005, and the claims were finally rejected in the Office Action mailed October 4, 2005.

VI. SUMMARY OF CLAIMED SUBJECT MATTER

The present Summary of Claimed Subject Matter includes background information in support of the claims, which is set forth immediately below, followed by a summary of each independent claim, including reference to Appellants' specification by page and line number, and reference to Appellants' drawings, which begins on page 8 of the present Brief.

A. Background Information

The present invention generally relates to assignment of internet protocol addresses and, more particularly, to automatically assigning internet protocol address information to a network device, such as a low-cost network adapter.

Referring to Appellants' Fig. 1 and their specification at page 3, lines 12-28, there is there is shown networked imaging system 10 that includes a computer 12, a networked device 14 and a network 16. Computer 12 includes software identified as a printer driver 18 and an operating system 20. Printer driver 18 and operating system 20 are communicatively interconnected. Networked device 14 may be an imaging device, such as a printer. In the embodiment of the invention described, networked device 14 will be in the form of a printer. Networked device 14 includes printer firmware 22 and a low-cost network adapter (LCNA) 24, which are communicatively interconnected. All network traffic directed to networked device 14 flows through LCNA 24 to printer firmware 22. Printer firmware 22 is responsible for generating a printed page on networked device 14, and printer firmware 22 relies on LCNA 24 to deliver printer control information and print data thereto. Network 16, such as a LAN, provides communicative interconnection between computer 12 and networked device 14 and other devices connected thereto which may or may not contain LCNAs.

Referring to Appellants' Fig. 1 and their specification at page 3, line 31 to page 4, line 2, printer driver 18 includes a data generation component 26, a printer driver user interface 28 and low-cost network adapter (LCNA) host software 30. Printer driver 18 contains the algorithms for assigning IP addresses, and more particularly, for automatically assigning an IP address to LCNA 24. Data generation component 26 generates data to be sent to networked device 14.

Referring to Appellants' Fig. 1 and their specification at page 4, lines 22-27, LCNA host software 30 communicates with IP stack 34 to obtain the IP address for networked device 14. If no IP address is available for networked device 14, then LCNA host software 30 is responsible for discovering LCNA 24 equipped devices on network 16. LCNA host software 30 configures LCNA 24 equipped devices, when appropriate, and provides a print connection over which data can be sent to networked device 14 through LCNA 24.

Referring to Appellants' specification at page 4, lines 28-31, LCNA 24 does not contain a mechanism for obtaining an IP address. Therefore, LCNA 24 depends on the operation of LCNA host software 30 on computer 12 to provide IP information thereto. LCNA 24 may be implemented as an application specific integrated circuit (ASIC).

B. Claims

Claim 1. Referring to Appellants' Fig. 1 and their specification at page 3, lines 12-28, a method of automatically assigning an internet protocol address to a device 14 includes providing a network 16, providing a computer 12 communicatively coupled to the network 16; providing a network adapter LCNA 24 to communicatively couple device 14 to network 16, network 16 providing communicative interconnection between computer 12 and network adapter LCNA 24.

Referring now to Fig. 2, computer 12 performs the steps of generating an internet protocol address (Step S108; Spec at page 6, lines 10-13); incorporating the internet protocol address in an address resolution protocol probe; and sending the address resolution protocol probe on network 16 (Step S110; Spec at page 6, lines 13-15); and determining whether a response to the address resolution protocol probe indicates that the internet protocol address is in use (Step S112; Spec at page 6, lines 15-17), wherein if the internet protocol address is not in use, then performing the step of assigning the internet protocol address to the network adapter LCNA 24 via network 16 (Step S118; Spec at page 6, lines 17-18 and page 7, lines 18-24).

Claim 11. Referring to Appellants' Fig. 1 and their specification at page 3, lines 12-28, a method of automatically assigning an internet protocol address to a device 14 includes providing a network 16; providing a computer 12 communicatively coupled to network 16; and providing a low-cost network adapter LCNA 24 to communicatively couple device 14 to network 16, network 16 providing communicative interconnection between computer 12 and low-cost network adapter LCNA 24.

Referring now to Fig. 2, computer 12 performs the steps of broadcasting a discovery packet on network 16 (Step S100, Spec at page 5, lines 13-18); receiving a response from low-cost network adapter LCNA 24 (Step S102, Spec at page 5, lines 18-24); determining if low-cost network adapter LCNA 24 has a valid internet protocol address (Step S104, Spec at page 5, lines 25-32).

If low-cost network adapter LCNA 24 does not have a valid internet protocol address, computer 12 then performs the steps of generating an internet protocol address (Step S108; Spec at page 6, lines 10-13); incorporating the internet protocol address in an address

resolution protocol probe; and sending the address resolution protocol probe on network 16 (Step S110; Spec at page 6, lines 13-15); and determining whether a response to the address resolution protocol probe indicates that the internet protocol address is in use (Step S112; Spec at page 6, lines 15-17), wherein if the internet protocol address is not in use, then performing the step of assigning the internet protocol address to low-cost network adapter LCNA 24 via network 16 (Step S118; Spec at page 6, lines 17-18 and page 7, lines 18-24).

Claim 17. Referring to Appellants' Fig. 1 and their specification at page 3, lines 12-28, a network based imaging system 10 includes a network 16; a computer 12 communicatively coupled to network 16; an imaging device 14; and a network adapter LCNA 24 communicatively coupling imaging device 14 to network 16, network 16 providing communicative interconnection between computer 12 and network adapter 16.

Referring now to Fig. 2, computer 12 executes instructions which generate an internet protocol address (Step S108; Spec at page 6, lines 10-13), incorporate the internet protocol address into an address resolution protocol probe, send the address resolution protocol probe on network 16 (Step S110; Spec at page 6, lines 13-15), utilize a response to the address resolution protocol probe to determine if the internet protocol address is in use (Step S112; Spec at page 6, lines 15-17), and if the internet protocol address is not in use, then assign the internet protocol address to network adapter LCNA 24 via network 16 (Step S118; Spec at page 6, lines 17-18 and page 7, lines 18-24).

VII. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1, 9-10, 17, and 25 were rejected under 35 U.S.C. §103(a) as being obvious over Buse, et al., U.S. Patent No. 6,810,420 B1 in view of Cheshire, S., Current Meeting Report, Cheshire, et al., 03/99.

B. Claims 2-6 and 18-22 were rejected under 35 U.S.C. §103(a) as being obvious over Buse in view of Cheshire, and in further view of Reed, et al., U.S. Patent No. 6,061,739.

C. Claims 7, 11-16, and 23 were rejected under 35 U.S.C. §103(a) as being unpatentable over Buse in view of Cheshire, and in further view of Mellquist, U.S. Patent No. 6,115,545.

D. Claims 8 and 24 were rejected under 35 USC §103(a) as being unpatentable over Buse in view of Cheshire, in further view of Mellquist, and in further view of Troll, Request for Comments: 2563, May 1999, Troll R.

VIII. ARGUMENT

A. CLAIMS 1, 9-10, 17, AND 25 ARE PATENTABLE UNDER 35 U.S.C. 103(a)

In the Final Office Action dated October 4, 2005, claims 1, 9, 10, 17, and 25 were rejected under 35 U.S.C. §103(a) as being unpatentable over Buse, et al., U.S. Patent No. 6,810,420 B1 (hereinafter, Buse) in view of Cheshire, S., Current Meeting Report, Cheshire, et al., 03/99 (hereinafter, Cheshire).

However, in determining whether obviousness is established by combining the teachings of the prior art, "the test is what the combined teachings of the references would have suggested to those of ordinary skill in the art," and the combined teachings of the prior art references must suggest, expressly or by implication, the improvements embodied by the invention. *In re GPAC Inc.* 35 USPQ2d 1116, 1123 (Fed Cir. 1995).

However, as set forth below, Appellants submit that claims 1, 9, 10, 17, and 25 are not disclosed, taught, or suggested by Buse in view of Cheshire, and are therefore patentable in their present form.

1. BUSE

Buse discloses a discovery scheme which can be operated by a proxy device such as a personal computer coupled to a local area network, and which facilitates the discovery of devices which may or may not be configured with an IP address (col. 1, lines 39-42). The discovery protocol performed by the proxy employs three basic packets, and when the proxy has resolved an IP address for the device, it sends an IP address allocated to the device, which then configures itself with the supplied parameters, and sends an "I_AM_HERE" frame with the address field being set to the allocated IP address (col. 2, lines 22-58, Figs. 2 and 3).

In order to resolve an IP address for the device, the proxy first sends a DHCP request, and if a DHCP server is available, that server provides a DHCP response including an IP address (col. 3, lines 23-26). If there is no DHCP response, the proxy allocates an IP address using Automatic Private IP addressing (col. 3, lines 28-37), and may verify that there is no address conflict using address resolution protocol or an ICMP echo request (col. 3, lines 37-41).

Thus, Buse provides to a device an IP address obtained via either a DHCP request or Automatic Private IP addressing.

2. CHESHIRE

Cheshire discloses automatic IP address assignment for a link local address with IPv4 (page 1), specifically the IPv4 self-configuration as currently implemented by Apple and MS (bottom paragraph of page 2). Operation as implemented in Mac OS 8.5 includes using a DHCP discover, and if no DHCP server is discovered, picking a random address, sending an ARP probe to verify that the address is not already in use, and if the address is in use, iterating the picking and repeating steps 10 times at most, otherwise configuring the computer's interface with the IP address (page 3).

Thus, Cheshire discloses a computer obtaining for itself an IP address via either a DHCP discover request or by picking a random address, verifying that it is not in use via an APR probe, and configuring that computer's interface with the IP address.

3. CLAIM 1 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE

Appellants' claim 1 is directed to a method of automatically assigning an internet protocol address to a device. Claim 1 recites, in part, providing a network; providing a computer communicatively coupled to said network; providing a network adapter to

communicatively couple said device to said network, said network providing communicative interconnection between said computer and said network adapter.

Claim 1 also recites said computer performing the steps of: generating an internet protocol address; incorporating said internet protocol address in an address resolution protocol probe; sending said address resolution protocol probe on said network; and determining whether a response to said address resolution protocol probe indicates that said internet protocol address is in use; wherein if said internet protocol address is not in use, then performing the step of assigning said internet protocol address to said network adapter via said network.

In contrast to claim 1, Buse discloses (1) obtaining an IP address via either a DHCP request or Automatic Private IP addressing, wherein (2) it may be verified that there is no address conflict using address resolution protocol or an ICMP echo request (col. 3, lines 37-41), and (3) providing the device with an address (col. 3, lines 23-41).

Although the Buse invention may arguably provide a device with an IP address, it does not do so by (1) generating an IP address; (2) incorporating the IP address in an ARP probe; (3) sending the ARP probe on the network; (4) determining whether a response to the ARP probe indicates that the IP address is in use; and (5), assigning the IP address to the network adapter via the network if the internet protocol address is not in use, as recited in claim 1.

Accordingly, assuming arguendo that Buse does provide a device with an IP address, Buse does not do so in a manner as recited in claim 1. Rather, the Buse approach is distinctly different from Appellants' invention of claim 1; the steps taken in the Buse disclosure are not common to claim 1, and clearly do not disclose, teach, or suggest

generating an internet protocol address; incorporating the internet protocol address in an address resolution protocol probe; sending the address resolution protocol probe on the network; and determining whether a response to the address resolution protocol probe indicates that the internet protocol address is in use; wherein if the internet protocol address is not in use, then performing the step of assigning the internet protocol address to the network adapter via the network, as recited in claim 1.

Appellants respectfully submit that the Patent and Trademark Office determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction "in light of the specification as it would be interpreted by one of ordinary skill in the art." *In re Am. Acad. of Sci. Tech. Ctr.*, 367 F.3d 1359, 1364 [70 USPQ2d 1827] (Fed. Cir. 2004) (Emphasis added).

Appellants respectfully direct the Board's attention to Appellants' specification at page 1, line 29 to page 2, line 14, which is reproduced as follows:

> There are several industry standards by which a network device can automatically obtain an IP address information. Such standards include the aforementioned DHCP, Universal Plug and Play (UPnP) and other forms of Automatic Private IP Addressing (APIPA). Each of these standards require that significant network transactions be initiated and conducted by the network device itself which requires hardware and configuration storage, making them cost prohibitive for low-cost devices.
>
> What is needed in the art is an apparatus and a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting traditional address assignment protocols.
>
> The present invention provides an apparatus and a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting the traditional address assignment protocols, such as DHCP, within the devices themselves. (Emphasis added).

Thus, Appellants claims may not be properly interpreted to encompass a method of obtaining an IP address via either a DHCP request or Automatic Private IP addressing, as disclosed by Buse.

Rather, Appellants' claimed invention is directed to a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting the traditional address assignment protocols, such as DHCP, within the devices themselves.

The Examiner acknowledges that Buse does not disclose, teach, or suggest the use of an ARP probe "nor where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network (Page 3 of Office Action mailed 10/4/05).

Rather, the Examiner relies upon Cheshire for as disclosing "where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network."

However, Cheshire simply does not make up for the deficiency of the Buse disclosure in rejecting claim 1.

For example, Cheshire discloses using a DHCP discover, and if no DHCP server is discovered, picking a random address, sending an ARP probe to verify that the address is not already in use, and if the address is in use, iterating the picking and repeating steps 10 times at most, otherwise configuring the computer's interface with the IP address (page 3).

However, the Cheshire disclosure specifically pertains to self-configuration (see bottom paragraph of page 2 of 7 of the Cheshire disclosure).

In addition, as set forth above, Appellants' claimed invention does not encompass a DHCP self-configuration approach to IP addressing.

Presently, Appellants' call the Board's attention to the fact that claim 1 recites that the (1) generating an IP address; (2) incorporating the IP address in an ARP probe; (3) sending the ARP probe on the network; (4) determining whether a response to the ARP probe indicates that the IP address is in use; and (5), assigning the IP address to the network adapter via the network if the internet protocol address is not in use, are performed relative to a network adapter that communicatively couples the device to the network, the network providing communicative interconnection between the computer and the network adapter, as recited in claim 1.

Thus, the network adapter of claim 1 is not associated with the computer that performs the assigning of the IP address, but rather, is in communication with the computer via the network.

In contrast, Cheshire is explicitly directed to self-configuration, which is known in the art as being a device that configures itself. In the case of Cheshire, one skilled in the art would clearly recognize that the steps taken by Cheshire are those steps taken in the MAC OS 8.5 for a computer to configure its own interface with an IP address, which is made explicit in the Cheshire disclosure on page 3 of 7.

Thus, Cheshire teaches self-configuration, as opposed to providing an IP address for another device, separate and distinct from the computer that obtains the IP address, which is connected via a network to the computer that obtains the IP address.

Accordingly, since Cheshire does not overcome the deficiency of Buse, as applied to claim 1, Buse and Cheshire, taken alone or in combination, do not disclose, teach, or suggest the subject matter of claim 1.

Although the Examiner asserts in the Response to Arguments that arguments against a reference individually cannot show nonobviousness where the rejections are based on a combination of references, Appellants respectfully submit that, as set forth above, the combination of Buse and Cheshire would not yield Appellants' claimed invention, since all of the limitations of claim 1 are not taught, disclosed, or suggested by Buse and Cheshire.

MPEP 2142 provides that to establish a prima facie case of obviousness, the prior art reference (or references when combined) must teach or suggest all the claim limitations.

Since Buse and Cheshire, taken alone or in combination, do not disclose, teach, or suggest all the limitations of claim 1, a prima facie case of obviousness has not been established against claim 1, and hence, claim 1 is allowable over Buse in view of Cheshire.

Notwithstanding the above, it would not be obvious to combine the teachings of Buse and Cheshire to achieve Appellants' invention of claim 1, i.e., to combine self-configuration, as disclosed by Cheshire, with the allocation of IP addresses by proxy, as disclosed by Buse, in order to achieve Appellants' claimed invention, at least for the reason that there would be no motivation to modify Buse with Cheshire, since each is a different approach in obtaining IP addresses.

Although the Examiner asserts that the motivation may be found in either the references themselves or knowledge generally available to one of ordinary skill in the art, relying upon In re Fine, 5 USPQ2d 1596 (Fed. Cir. 1988), the Examiner has not provided particular findings as to the reason the skilled artisan, with no knowledge of the claimed

invention, would have selected the particular components from Cheshire and Buse for combination in the manner claimed.

There was no specific understanding or principle within the knowledge of a skilled artisan that would have motivated one with no knowledge of Appellants' invention to make the combination in the manner of claim 1. *In re Kotzab,* 55 USPQ2d 1313, 1317 (Fed. Cir. 2000). See also *In Re Lee*, 277 F.3d 1338 (Fed. Cir. 2002).

In addition, prior art references in combination do not make the invention obvious unless something in the prior art references would suggest an advantage to be derived from combining their teachings. *In re Sernaker*, 217 USPQ 1 (Fed. Cir. 1983). See also *In re GPAC*, 35 USPQ2d 1116, 1123 (Fed. Cir. 1995).

Also, MPEP 2144 provides that the expectation of some advantage is the strongest rationale for combining references.

However, neither Buse nor Cheshire disclose, teach, or suggest an advantage to be derived from combining their teachings in the manner attempted in rejecting claim 1.

Further, Appellants submit that the asserted combination is based on impermissible hindsight reconstruction. It is impermissible to use the claimed invention as an instruction manual or "template" to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re Fritch*, 23 USPQ2d 1780, 1784 (Fed. Cir. 1992).

Buse is directed to configuration by proxy, whereas Cheshire is directed to self-configuration.

Since, as set forth above, there is no advantage disclosed, taught, or suggested by either of the references to modify Buse with Cheshire, and no asserted principle within the knowledge of a skilled artisan that would have motivated one with no knowledge of

Appellants' invention to make the combination, it seems clear that hindsight reconstruction has been employed in rejecting Appellants' claims.

Accordingly, claim 1 is not obvious over Buse in view of Cheshire.

Thus, for at least the reasons set forth above, Appellants submit that claim 1 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 1 under 35 U.S.C. 103(a).

4. CLAIMS 9 AND 10 ARE PATENTABLE OVER BUSE IN VIEW OF CHESHIRE

Claims 9 and 10 are believed allowable due to their dependence on otherwise allowable base claim 1. In addition, claims 9 and 10 further and patentably define Appellants' invention over Buse in view of Cheshire.

Thus, for at least the reasons set forth above, Appellants submit that claims 9 and 10 are patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claims 9 and 10 under 35 U.S.C. 103(a).

5. CLAIM 17 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE

Appellants' claim 17 is directed to a network based imaging system. Claim 17 recites, in part, wherein said computer executes instructions which generate an internet protocol address, incorporate said internet protocol address into an address resolution protocol probe, send said address resolution protocol probe on said network, utilize a response to said address resolution protocol probe to determine if said internet protocol

address is in use and if said internet protocol address is not in use, then assign said internet protocol address to said network adapter via said network.

Claim 17 is believed allowable for substantially the same reasons as set forth above with respect to claim 1.

Claim 17 also recites, in part, an imaging device; and a network adapter communicatively coupling said imaging device to said network, said network providing communicative interconnection between said computer and said network adapter.

Buse simply does not disclose, teach, or suggest an imaging device, and nor does the Examiner assert as much. In addition, although Cheshire offhandedly mentions "printers," (page 5), Cheshire does not disclose, teach, or suggest an imaging device; and a network adapter communicatively coupling the imaging device to the network, the network providing communicative interconnection between the computer and the network adapter.

MPEP 2142 requires that in order to establish a *prima facie* case of obviousness, all claim limitations must be taught or suggested by the prior art references.

Because Buse in view of Cheshire simply do not disclose, teach, or suggest all of the limitations of claim 17, e.g., an imaging device; and a network adapter communicatively coupling the imaging device to the network, the network providing communicative interconnection between the computer and the network adapter, Appellants invention of claim 17 is not obvious over Buse in view of Cheshire as per MPEP2142.

In rejecting claim 17, the Examiner asserts that "in the absence of an express intent to impart a novel meaning to the claim terms, the words are presumed to take on the ordinary and customary meanings attributed to them by one of ordinary skill in the art."

Further, the Examiner asserts that "The broadest reasonable interpretation has been applied to the claims as mandated, thereby, claimed 'imaging device', for the purposes of examination given the broadest reasonable interpretation is a device." (Emphasis added).

However, the Patent and Trademark Office determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction "in light of the specification as it would be interpreted by one of ordinary skill in the art." *In re Am. Acad. of Sci. Tech. Ctr.*, 367 F.3d 1359, 1364 [70 USPQ2d 1827] (Fed. Cir. 2004) (Emphasis added).

Appellants' specification provides that network device 14 may be an imaging device, such as a printer (p.3, l.20). In interpreting "imaging device" to include any "device" generally, the Examiner has truncated Appellants' chosen term, and simply removed the word, "imaging," along with the meaning that "imaging" imports to "imaging device." Interpreted in light of Appellants' specification, one of ordinary skill in the art would not interpret "imaging device" broadly enough to encompass any "device." Rather, an imaging device is known in the art to pertain to, e.g., a printer, a copier, an all-in-one unit that combines printing, copying, and faxing, etc., and the like.

Accordingly, Appellants respectfully submit that upon giving claim 1 its broadest reasonable construction in light of the specification as it would be interpreted by one of ordinary skill in the art, the claim 1 term, "imaging device," would not be interpreted by one skilled in the art as broadly as the general term, "device," as asserted by the Examiner.

Since Buse and Cheshire, taken alone or in combination, do not disclose, teach, or suggest all the limitations of claim 17, a case of prima facie obviousness has not been

established against claim 17, and hence, claim 17 is allowable over Buse in view of Cheshire.

In addition, it would not be obvious to combine Buse and Cheshire for substantially the same reasons as set forth above with respect to claim 1.

Accordingly, for at least the reasons set forth above, Appellants respectfully submit that claim 17 is patentable over the cited references, Buse in view of Cheshire.

Thus, for at least the reasons set forth above, Appellants submit that claim 17 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 17 under 35 U.S.C. 103(a).

6. CLAIM 25 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE

Claim 25 is believed allowable due to its dependence on otherwise allowable base claim 17.

Thus, for at least the reasons set forth above, Appellants submit that claim 25 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 25 under 35 U.S.C. 103(a).

B. CLAIMS 2-6 AND 18-22 ARE PATENTABLE UNDER 35 U.S.C. 103(a)

In the Final Office Action dated October 4, 2005, claims 2-6 and 18-22 were rejected under 35 U.S.C. §103(a) as being unpatentable over Buse in view of Cheshire, and in further view of Reed, et al., U.S. Patent No. 6,061,739 (hereinafter, Reed).

However, in determining whether obviousness is established by combining the teachings of the prior art, "the test is what the combined teachings of the references would

have suggested to those of ordinary skill in the art," and the combined teachings of the prior art references must suggest, expressly or by implication, the improvements embodied by the invention. *In re GPAC Inc.* 35 USPQ2d 1116, 1123 (Fed Cir. 1995).

However, as set forth below, Appellants submit that claims 2-6 and 18-22 are not disclosed, taught, or suggested by Buse in view of Cheshire, and in further view of Reed, and are therefore patentable in their present form.

1. BUSE

Buse is summarized at the beginning of section VIII(A)(1) of this Brief, which for the sake of brevity is not repeated here.

2. CHESHIRE

Cheshire is summarized at the beginning of section VIII(A)(2) of this Brief, which for the sake of brevity is not repeated here.

3. REED

Reed discloses a method for assigning a network address to a new device coupled to a network without any additional infrastructure or pre-existing knowledge of the hardware address of the device (col. 4, lines 19-22). The device attempts to establish a connection on the network, resulting in ARP requests being generated (col. 4, lines 22-25). The device monitors the communications on the network for unanswered ARP requests (col. 4, lines 25-27). When the device sees N unanswered ARP requests in a given length of time, it adopts the requested network address and responds to the ARP with its hardware address (col. 4, lines 27-30, Fig. 2).

Thus, the Reed device performs self-configuration.

4. CLAIMS 2-6 ARE PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF REED

Each of claims 2-6 depend directly or indirectly from claim 1. As set forth above with respect to claim 1, Buse and Cheshire, taken alone or in combination, would not yield the subject matter of claim 1, and the subject matter of claim 1 is not obvious over Buse in view of Cheshire.

Appellants respectfully submit that Reed does not overcome the deficiency of Buse in view of Cheshire, as applied to claim 1, nor does the Examiner assert as much.

In rejecting claim 1, the Examiner acknowledges that Buse does not disclose, teach, or suggest the use of an ARP probe, "nor where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network (Page 3 of Office Action mailed 10/4/05).

Rather, the Examiner relies upon Cheshire for as disclosing "where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network."

However, as set forth above with respect to claim 1, Cheshire simply does not make up for the deficiency of the Buse disclosure as applied to claim 1.

In addition, Reed does not make up for the deficiency of Buse and Cheshire as applied to claim 1.

Like Cheshire, Reed discloses self-configuration of an IP address.

For example, Reed et al, discloses that the device attempts to establish a connection, causing ARP requests to be generated, and when the device sees N unanswered ARP requests (where N is a preset threshold) in a given length of time, the device adopts the

requested network address and responds to the ARP with its hardware address (col. 4, lines 22-30, Fig. 2).

Thus, by sending and responding to ARP communications, the device configures itself with an IP address, in contrast to claim 1, wherein a computer performs the step of assigning the internet protocol address to the network adapter that communicatively couples the device to the network, via the network, i.e., the network adapter for the device is configured by the computer via the network.

Thus, since Reed does not overcome the deficiency of Buse and Cheshire as applied to claim 1, the combination of Buse, Cheshire, and Reed would not yield Appellants' claimed invention.

Although the Examiner asserts in the Response to Arguments that arguments against a reference individually cannot show nonobviousness where the rejections are based on a combination of references, Appellants respectfully submit that, as set forth above, the combination of Buse, Cheshire, and Reed would not yield Appellants' claimed invention, since all of the limitations of claim 1 are not taught, disclosed, or suggested by Buse, Cheshire, and Reed, taken alone or in combination.

MPEP 2142 provides that to establish a prima facie case of obviousness the prior art reference (or references when combined) must teach or suggest all the claim limitations.

Since Buse, Cheshire, and Reed, taken alone or in combination, do not disclose, teach, or suggest all the limitations of claim 1, claim 1 a case of prima facie obviousness has not been established against claim 1, and hence, claim 1 is allowable over Buse in view of Cheshire and in further view of Reed.

Claims 2-6 are thus believed allowable due to their dependence, directly or indirectly, on otherwise allowable base claim 1.

Notwithstanding the above, since Reed is directed to and discloses self-configuration, for substantially the same reasons as set forth above with respect to claim 1 regarding Buse in view of Cheshire, it would not have been obvious to combine the teachings of Cheshire and Reed, i.e., self-configuration, with the allocation of IP addresses by proxy, as disclosed by Buse.

For example, the mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the desirability of the modification. *In re Laskowski,* 10 USPQ2d 1397 (Fed. Cir. 1989). In addition, prior art references in combination do not make the invention obvious unless something in prior art references would suggest advantage to be derived from combining their teachings. *In re Sernaker,* 217 USPQ 1 (Fed. Cir. 1983).

Also, MPEP 2144 provides that the expectation of some advantage is the strongest rationale for combining references.

However, there is nothing disclosed, taught, or suggested in either of the Buse or Cheshire or Reed references that would suggest the desirability of the asserted combination or that there would be an advantage to be derived from their teachings. Thus, it would not have been obvious to combine the prior art references since there is simply nothing in those references suggests that their teachings could be successfully combined to yield advantageous results in the primary reference. *In re Sernaker,* 217 USPQ 1 (Fed. Cir. 1983).

In addition, Appellants submit that since Cheshire and Reed are directed to self-configuration, in contrast to Buse, one would not have been motivated to modify Buse with Cheshire and Reed, since Buse is directed to configuration by proxy, and also discloses what is purported to be an operational method to configure a device by proxy, that does not purport to need modification in order to achieve its desired result.

That is, there would be no motivation to modify a method for configuration by proxy that is operational in of itself (Buse), much less by incorporating aspects of a method for self-configuration (Cheshire and/or Reed).

Appellants further contend that the way in which the Examiner has assembled the combination of Buse, Cheshire and Reed, without any advantage disclosed, taught, or suggested by the references, is tantamount to impermissible hindsight reconstruction of Appellants' claims.

It is impermissible to use the claimed invention as an instruction manual or "template" to piece together the teachings of the prior art so that the claimed invention is rendered obvious. In re Fritch, (CA FC) 23 USPQ2d 1780, 1784 (Fed. Cir. 1992). For example, Buse is directed to configuration by proxy, whereas Cheshire and Reed are directed to self-configuration.

Since there is no advantage disclosed, taught, or suggested by any of the references (Buse, Cheshire, and/or Reed) to modify Buse with Cheshire or Reed, it follows that there would not be a motivation to combine the references, and consequently, Appellants submit that the asserted combination is based on impermissible hindsight reconstruction.

Accordingly, Appellants respectfully submit that it would not be obvious to modify Buse with Cheshire and Reed, and thus, claims 2-6, depending from claim 1, are not unpatentable over Buse in view of Cheshire and in further view of Reed.

Thus, for at least the reasons set forth above, Appellants submit that claims 2-6 are patentable in their present respective forms.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claims 2-6 under 35 U.S.C. 103(a).

5. CLAIMS 18-22 ARE PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF REED

Claims 18-22 were rejected on the same basis as claims 2-6, and are believed allowable for substantially the same reasons as set forth above with respect to claims 2-6.

Thus, for at least the reasons set forth above, Appellants submit that claims 18-22 are patentable in their present respective forms.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claims 18-22 under 35 U.S.C. 103(a).

C. CLAIMS 7, 11-16, AND 23 ARE PATENTABLE UNDER 35 U.S.C. 103(a)

In the Final Office Action dated October 4, 2005, claims 7, 11-16, and 23 were rejected under 35 U.S.C. §103(a) as being unpatentable over Buse in view of Cheshire, and in further view of Mellquist, U.S. Patent No. 6,115,545 (hereinafter, Mellquist).

However, in determining whether obviousness is established by combining the teachings of the prior art, "the test is what the combined teachings of the references would have suggested to those of ordinary skill in the art," and the combined teachings of the prior

art references must suggest, expressly or by implication, the improvements embodied by the invention. *In re GPAC Inc.* 35 USPQ2d 1116, 1123 (Fed Cir. 1995).

However, as set forth below, Appellants submit that claims 7, 11-16, and 23 are not disclosed, taught, or suggested by Buse in view of Cheshire, and in further view of Mellquist, and are therefore patentable in their present form.

1. BUSE

Buse is summarized at the beginning of section VIII(A)(1) of this Brief, which for the sake of brevity is not repeated here.

2. CHESHIRE

Cheshire is summarized at the beginning of section VIII(A)(2) of this Brief, which for the sake of brevity is not repeated here.

3. MELLQUIST

Mellquist discloses as background, the use of a BOOTstrap Protocol (BOOTP) that allows clients to automatically receive all IP configuration information from a configured BOOTP server (col. 2, lines 26-30). In order to define an IP address, a free address in the range of valid addresses must be selected (col. 3, lines 12-14). Addresses are usually administered by a person who allocates these addresses to entities who require them (col. 3, lines 14-15). It is important that duplicate addresses are not allowed since this can cause major trouble (col. 3, lines 16-17). Also, a sub-net mask is required for proper operation, and must be the same on all entities across the sub-net (col. 3, lines 17-19).

The Mellquist apparatus includes a configuration module 41 that acts in place of a BOOTP server to accept and reply to a select set of BOOTP requests from devices, wherein the BOOTP response contains an IP address corresponding to a media access control (MAC)

address for the device that submitted the BOOTP request (col. 5, lines 36-45). Once powered up, a network device 33 issues a broadcast BOOTP request 47 which will be picked up by IP configuration module 41, that issues a BOOTP response 48 by which network device 33 will obtain the IP configuration parameters and proceed to initialize (col. 5, line 66 to col. 6, line 5).

4. CLAIM 7 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF MELLQUIST

Claim 7 is directed to the method of claim 1, wherein prior to performing said generating step. Claim 7 recites, broadcasting a discovery packet on said network; receiving a response from said network adapter; and determining if said network adapter has a valid internet protocol address.

As set forth above with respect to claim 1, Buse and Cheshire, taken alone or in combination, would not yield the subject matter of claim 1, and the subject matter of claim 1 is not obvious over Buse in view of Cheshire.

Appellants respectfully submit that Mellquist does not overcome the deficiency of Buse in view of Cheshire, as applied to claim 1, nor does the Examiner assert as much.

Claim 7 is thus believed allowable due to its dependence on otherwise allowable base claim 1.

For example, in rejecting claim 1, the Examiner acknowledges that Buse does not disclose, teach, or suggest the use of an ARP probe "nor where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network (Page 3 of Office Action mailed 10/4/05).

Rather, the Examiner relies upon Cheshire for as disclosing "where assigning an IP address to a device is performed by assigning the IP address to the network adapter of the device which connects the device to the network."

However, as set forth above with respect to claim 1, Cheshire simply does not make up for the deficiency of the Buse disclosure as applied to claim 1.

In addition, Mellquist does not make up for the deficiency of Buse and Cheshire as applied to claim 1.

Like Cheshire, Mellquist discloses self-configuration of an IP address.

For example, Mellquist et al, discloses that a network device 33 sends out a BOOTP request, and IP configuration module 41, standing in the place of a BOOTP server, provides a BOOTP response including an IP address to network device 33, which then proceeds to initialize. Thus network device 33 configures itself by obtaining an IP address from IP configuration module 41 that acts in the place of a BOOTP server, a process which is known in the art to be self-configuration.

In contrast to a network device that configures itself based on submitting a BOOTP request and receiving a BOOTP response, as disclosed by Mellquist, claim 1 contemplates a computer that performs the step of assigning the internet protocol address to the network device, i.e., a network adapter associated with a device other than the computer that assigns the IP address, via the network.

Unlike the Mellquist disclosure, Appellants' invention allows the use of a network adapter that is unable to configure itself, i.e., a network adapter that does not contain a mechanism for obtaining an IP address, and depends on another computer to do so (see Appellants' specification at page 4, lines 28-31).

Thus, since Mellquist does not overcome the deficiency of Buse and Cheshire as applied to claim 1, the combination of Buse, Cheshire, and Mellquist would not yield Appellants' claimed invention.

Although the Examiner asserts in the Response to Arguments that arguments against a reference individually cannot show nonobviousness where the rejections are based on a combination of references, Appellants respectfully submit that, as set forth above, the combination of Buse, Cheshire, and Mellquist would not yield Appellants' claimed invention, since all of the limitations of claim 1 are not taught, disclosed, or suggested by Buse, Cheshire, and Mellquist, taken alone or in combination.

MPEP 2142 provides that to establish a prima facie case of obviousness the prior art reference (or references when combined) must teach or suggest all the claim limitations.

Since Buse, Cheshire, and Mellquist, taken alone or in combination, do not disclose, teach, or suggest all the limitations of claim 1, claim 1 a case of prima facie obviousness has not been established against claim 1, and hence, claim 1 is allowable over Buse in view of Cheshire and in further view of Mellquist.

Claim 7 is thus believed allowable due to its dependence, directly or indirectly, on otherwise allowable base claim 1.

In addition, claim 7 recites, in part, determining if the network adapter has a valid internet protocol address.

In rejecting claim 7, the Examiner relies on Mellquist at column 3, lines 11-19.

Appellants respectfully submit that the relied-upon language of Mellquist merely discloses that a required free address in the range of valid addresses must be selected (col. 3, lines 12-14), that addresses are usually administered by a person who allocates these

addresses to entities who require them (col. 3, lines 14-15), and that duplicate addresses are not allowed (col. 3, lines 16-17).

However, such language simply does not disclose, teach, or suggest any "determination" aspect, much less determining if the network adapter has a valid internet protocol address, as recited in claim 7.

In the Response to Arguments, the Examiner asserts that "The broadest reasonable interpretation has been applied to the claim term "valid internet protocol address."

However, Appellants respectfully submit that the Patent and Trademark Office determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction "in light of the specification as it would be interpreted by one of ordinary skill in the art." *In re Am. Acad. of Sci. Tech. Ctr.*, 367 F.3d 1359, 1364 [70 USPQ2d 1827] (Fed. Cir. 2004) (Emphasis added).

In order to clarify determining if the network adapter has a valid internet protocol address, as recited in claim 7, Appellants respectfully direct the Board's attention to Appellants' specification at page 5, lines 25-32, which is reproduced as follows:

> At step 104, computer 12 evaluates the response from LCNA 24 to determine if LCNA 24 has a valid IP address. An IP address is considered valid if it is an appropriate address for the subnet to which computer 12 is connected. An uninitialized LCNA always has an invalid IP address. The determination of validity is accomplished by comparing the value associated with the IP address of LCNA 24 to the IP address of computer 12 and a subnet mask of computer 12. If the IP address is valid, then the process terminates at step 120. Otherwise, the process flow continues at step 106.

Mellquist simply does not disclose, teach, or suggest determining if the network adapter has a valid internet protocol address. For example, the relied-upon Mellquist statements merely lists two existing constraints on IP addresses, and disclose that addresses

are administered by a person who allocates the addresses, without stating that there is a determination as to whether an address is valid.

The relied upon Mellquist text simply does not disclose, teach, or suggest finding out if the address is valid by investigation, comparison with known values, reasoning, or calculation, as would constitute determining if the network adapter has a valid internet protocol address as would be interpreted by one of ordinary skill in the art in light of Appellants' specification.

Rather, the relied upon text simply indicates that a free address must be used, indicates who usually provides the addresses, and indicates that duplicate addresses are not allowed, without a determining aspect within the context of Appellants' claimed invention.

Although Buse discloses checking for an IP address conflict, and Cheshire discloses determining whether an address is already in use, neither Buse nor Cheshire disclose, teach, or suggest determining if the network adapter has a valid internet protocol address in the context of Applicants' claimed invention.

Thus, Buse, Cheshire, and Mellquist, taken alone or in combination, do not disclose, teach, or suggest determining if the network adapter has a valid internet protocol address as would be interpreted by one of ordinary skill in the art in light of Appellants' specification, and hence, the combination of Buse, Cheshire, and Mellquist would not yield Appellants' claimed invention.

Accordingly, claim 7 is believed allowable in its own form.

Notwithstanding the above, since Mellquist discloses self-configuration, for substantially the same reasons as set forth above with respect to claim 1 regarding Buse in view of Cheshire, it would not have been obvious to combine the teachings of Cheshire and

Mellquist, i.e., self-configuration, with the allocation of IP addresses by proxy, as disclosed by Buse.

For example, the mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the desirability of the modification. In re Laskowski 10 USPQ2d 1397 (Fed. Cir. 1989). In addition, prior art references in combination do not make the invention obvious unless something in prior art references would suggest advantage to be derived from combining their teachings. In re Semaker 217 USPQ 1 (Fed. Cir. 1983).

Also, MPEP 2144 provides that the expectation of some advantage is the strongest rationale for combining references.

However, there is nothing disclosed, taught, or suggested in either of the Buse or Cheshire or Mellquist references that would suggest the desirability of the asserted combination or that there would be an advantage to be derived from their teachings. Thus, it would not have been obvious to combine the prior art references since there is simply nothing in those references suggests that their teachings could be successfully combined to yield advantageous results in the primary reference. In re Semaker 217 USPQ 1 (Fed. Cir. 1983).

In addition, Appellants submit that since Cheshire and Mellquist are directed to self-configuration, in contrast to Buse, one would not have been motivated to modify Buse, since Buse is directed to configuration by proxy, and also discloses what is purported to be operational method to perform a configuration for a device by proxy that does not purport to need modification in order to achieve its desired result. That is, there would be no motivation to modify a method for configuration by proxy that is operational in of itself

(Buse), much less by incorporating aspects of a method for self-configuration (Cheshire and/or Mellquist).

Appellants further contend that the way in which the Examiner has assembled the combination of Buse, Cheshire and Mellquist, without any advantage disclosed, taught, or suggested by the references, is tantamount to impermissible hindsight reconstruction of Appellants' claims.

It is impermissible to use the claimed invention as an instruction manual or "template" to piece together the teachings of the prior art so that the claimed invention is rendered obvious. In re Fritch, (CA FC) 23 USPQ2d 1780, 1784 (Fed. Cir. 1992). For example, Buse is directed to configuration by proxy, whereas Cheshire and Mellquist are directed to self-configuration.

Since there is no advantage disclosed, taught, or suggested by any of the references (Buse, Cheshire, and/or Mellquist) to modify Buse with Cheshire or Mellquist, it follows that there would not be a motivation to combine the references, and consequently, Appellants submit that the asserted combination is based on impermissible hindsight reconstruction.

Accordingly, Appellants respectfully submit that it would not be obvious to modify Buse with Cheshire and Reed, and thus, claim 7, depending from claim 1, is not unpatentable over Buse in view of Cheshire and in further view of Mellquist.

Thus, for at least the reasons set forth above, Appellants submit that claim 7 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 7 under 35 U.S.C. 103(a).

5. CLAIM 11 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF MELLQUIST

Claim 11 is directed to a method of automatically assigning an internet protocol address to a device. Claim 11 recites, in part, determining if said low-cost network adapter has a valid internet protocol address. For substantially the same reasons as set forth above with respect to claim 7, Appellants respectfully submit that Buse in view of Cheshire, and in further view of Mellquist does not disclose, teach, or suggest determining if the low-cost network adapter has a valid internet protocol address.

Claim 11 also recites, in part, the computer performing the steps of: generating an internet protocol address; incorporating said internet protocol address in an address resolution protocol probe; sending said address resolution protocol probe on said network; and determining whether a response to said address resolution protocol probe indicates that said internet protocol address is in use; wherein if said internet protocol address is not in use, then performing the step of assigning said internet protocol address to said low-cost network adapter via said network.

Claim 11 is believed allowable over Buse in view of Cheshire for substantially the same reasons set forth above with respect to claims 1 and 7, since as set forth above with respect to claim 7, Mellquist does not overcome the deficiency of Buse and Cheshire as applied to claim 1, nor does the Examiner assert as much.

Accordingly, for at least the reasons set forth above, Buse in view of Cheshire, and in further view of Mellquist, taken alone or in combination, do not disclose, teach, or suggest the subject matter of claim 11, and the combination of Buse, Cheshire, and Mellquist would not yield Appellants invention of claim 11.

Thus, for at least the reasons set forth above, Appellants submit that claim 11 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 11 under 35 U.S.C. 103(a).

6. CLAIMS 12-16 ARE PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF MELLQUIST

Each of claims 12-16 depend directly or indirectly from claim 11. As set forth above, Buse, Cheshire, and Mellquist, taken alone or in combination, would not yield the subject matter of claim 11, and the subject matter of claim 11 is not obvious over Buse in view of Cheshire.

Accordingly, claims 12-16 are believed allowable due to their dependence on otherwise allowable base claim 11.

Thus, for at least the reasons set forth above, Appellants submit that claims 12-16 are patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claims 12-16 under 35 U.S.C. 103(a).

7. CLAIM 23 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, AND IN FURTHER VIEW OF MELLQUIST

Claim 23 is directed to the system of claim 17. Claim 23 recites, in part, wherein the computer executes preliminary instructions to determine if said network adapter has a valid internet protocol address. For substantially the same reasons as set forth above with respect to claim 7, Buse in view of Cheshire, and in further view of Mellquist do not disclose, teach, or suggest wherein the computer executes preliminary instructions to determine if the

network adapter has a valid internet protocol address, as recited in claim 23. Accordingly, claim 23 is believed allowable in its present form.

In addition, claim 23 is believed allowable due to its dependence on otherwise allowable base claim 17.

Thus, for at least the reasons set forth above, Appellants submit that claim 23 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 23 under 35 U.S.C. 103(a).

D. CLAIMS 8 AND 24 ARE PATENTABLE UNDER 35 U.S.C. 103(a)

In the Final Office Action dated October 4, 2005, claims 8 and 24 were rejected under 35 U.S.C. §103(a) as being unpatentable over Buse in view of Cheshire, in further view of Mellquist, and in further view of Troll, Request for Comments: 2563, May 1999, Troll R. (hereinafter, Troll).

However, in determining whether obviousness is established by combining the teachings of the prior art, "the test is what the combined teachings of the references would have suggested to those of ordinary skill in the art," and the combined teachings of the prior art references must suggest, expressly or by implication, the improvements embodied by the invention. *In re GPAC Inc.* 35 USPQ2d 1116, 1123 (Fed Cir. 1995).

However, as set forth below, Appellants submit that claims 8 and 24 are not disclosed, taught, or suggested by Buse in view of Cheshire, in further view of Mellquist, and in further view of Troll, and are therefore patentable in their present form.

1. BUSE

Buse is summarized at the beginning of section VIII(A)(1) of this Brief, which for the sake of brevity is not repeated here.

2. CHESHIRE

Cheshire is summarized at the beginning of section VIII(A)(2) of this Brief, which for the sake of brevity is not repeated here.

3. MELLQUIST

Mellquist is summarized at the beginning of section VIII(C)(3) of this Brief, which for the sake of brevity is not repeated here.

4. TROLL

Troll is directed to disabling stateless auto-configuration in IPv4 clients (page 1), and allowing a DHCP client to determine whether or not it should assign itself a "link-local" address (page 2). Troll also discloses an auto-configure option which allows a DHCP client to determine whether or not it should generate a link-local IP address.

5. CLAIM 8 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, IN FURTHER VIEW OF MELLQUIST, AND IN FURTHER VIEW OF TROLL

Claim 8 is directed to the method of claim 7, wherein prior to performing said generating step said method comprising the step of determining whether said network allows said computer to assign an internet protocol address to said network adapter.

Claim 8 depends from claim 7, which depends from claim 1. As set forth above with respect to claim 7, the subject matter of either of claims 1 and 7 is not unpatentable over Buse in view of Cheshire, and in further view of Mellquist. Appellants respectfully submit

that Troll does not overcome the deficiency of Buse in view of Cheshire, and in further view of Mellquist, nor does the Examiner assert as much.

For example, as set forth above, and as acknowledged by the Examiner, Troll is directed to a DHCP client assigning itself an IP address, which is known in the art as self assignment, or self configuration.

In contrast, however, claims 1 and 7 contemplate a computer that performs the step of assigning the internet protocol address to the network device, i.e., the network adapter, via the network, which allows the use of a network adapter that is unable to configure itself, i.e., a network adapter that does not contain a mechanism for obtaining an IP address, and depends on another computer to do so (see Appellants' specification at page 4, lines 28-31).

As set forth above with respect to claims 1 and 7, Buse in view of Cheshire and in further view of Mellquist would not yield Appellants' invention of claim 7, and that it would not be obvious to combine the teaching of Buse in view of Cheshire.

For substantially the same reasons as set forth above with respect to claims 1 and 7 as with respect to Buse in view of Cheshire, it would not have been obvious to combine the teachings of Cheshire, Mellquist, and Troll, i.e., self-configuration, with the allocation of IP addresses by proxy, as disclosed by Buse.

Accordingly, claim 8 is not unpatentable over Buse in view of Cheshire, in further view of Mellquist, and in further view of Troll, due to its dependence on otherwise allowable base claim 1 and intervening claim 7.

In addition, Troll does not disclose, teach, or suggest determining whether the network allows the computer to assign an internet protocol address to the network adapter, as recited in claim 8, nor do the other cited references.

Rather, Troll discloses that a DHCP client will be able to determine whether the network is centrally administered, thus allowing it to determine whether or not it should assign itself an address (page 2).

Troll also discloses that a DHCP client will be allowed to determine whether or not it should generate an address (page 3). However, the relied-upon Troll disclosures have no bearing on and do not disclose, teach, or suggest determining whether the network allows the computer to assign an internet protocol address to the network adapter, as recited in claim 8.

In the Response to Arguments, the Examiner asserts that "The broadest reasonable interpretation has been applied to the claims, and that "a network that allows a computer to assign an internet protocol address to a network adapter" "simply means determining that DHCP server(s) are available on the network from which remote automatic IP addresses can be obtained for providing to a devices' network adapter."

However, Appellants respectfully submit that the Patent and Trademark Office determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction "in light of the specification as it would be interpreted by one of ordinary skill in the art." *In re Am. Acad. of Sci. Tech. Ctr.*, 367 F.3d 1359, 1364 [70 USPQ2d 1827] (Fed. Cir. 2004) (Emphasis added).

In order to clarify determining whether the network allows the computer to assign an internet protocol address to the network adapter, as recited in claim 8, Appellants respectfully direct the Board's attention to Appellants' specification at page 5, line 33 to page 6, line 9 which is reproduced as follows:

> At step 106, computer 12 determines if network 16 allows automatic remote assignment of IP addresses. If network 16 allows automatic remote assignment of IP addresses, then process flow continues at step 108. Otherwise, the process terminates at step 120. Computer 12 provides for the

> manual assignment of an IP address, which is not a part of this invention, thus in the event network 16 does not allow automatic remote assignment of IP addresses, an IP address can be assigned manually.
>
> Determination as to whether network 16 allows the assignment of IP addresses to LCNA type devices is necessary since some network environments do not allow for automatic remote IP address assignment. If the network environment utilizes certain addresses, such as those used by the UPnP or APIPA addressing schemes, then automatic remote IP address assignment is possible.

Appellants further respectfully direct the Board's attention to Appellants' specification at page 1, line 29 to page 2, line 14 which is reproduced as follows:

> There are several industry standards by which a network device can automatically obtain an IP address information. Such standards include the aforementioned DHCP, Universal Plug and Play (UPnP) and other forms of Automatic Private IP Addressing (APIPA). Each of these standards require that significant network transactions be initiated and conducted by the network device itself which requires hardware and configuration storage, making them cost prohibitive for low-cost devices.
>
> What is needed in the art is an apparatus and a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting traditional address assignment protocols.
>
> The present invention provides an apparatus and a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting the traditional address assignment protocols, such as DHCP, within the devices themselves. (Emphasis added).

Appellants respectfully submit that upon giving claim 8 its broadest reasonable construction "in light of the specification as it would be interpreted by one of ordinary skill in the art," one of ordinary skill in the art would not interpret "a network that allows a computer to assign an internet protocol address to a network adapter" to "simply mean[s] determining that DHCP server(s) are available on the network from which remote automatic IP addresses can be obtained for providing to a devices' network adapter," as asserted by the examiner.

For example, Appellant's specification is clearly directed to a method by which a device on a computer network can be assigned an IP address automatically, without the overhead of supporting the traditional address assignment protocols, such as DHCP, within the devices themselves (Spec at page 2, lines 11-14).

Appellants respectfully submit that the fact of the Examiner's reliance on a combination of 4 references, in the manner set forth to reject claim 8, supports Appellants' present contention of impermissible hindsight reconstruction of Appellants invention, using Appellants' claims as a blueprint, for at least the reasons set forth above with respect to claim 1.

Thus, for at least the reasons set forth above, Appellants submit that claim 8 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 8 under 35 U.S.C. 103(a).

6. CLAIM 24 IS PATENTABLE OVER BUSE IN VIEW OF CHESHIRE, IN FURTHER VIEW OF MELLQUIST, AND IN FURTHER VIEW OF TROLL

Claim 24 was rejected on the same basis as claim 8, and is believed allowable for substantially the same reasons as set forth above with respect to claims 8.

Thus, for at least the reasons set forth above, Appellants submit that claim 24 is patentable in its present form.

Accordingly, Appellants respectfully request that the Board reverse the rejection of claim 24 under 35 U.S.C. 103(a).

E. CONCLUSION

For the foregoing reasons, Appellants submit that claims 1-25 are patentable in their present respective forms. Accordingly, Appellants respectfully requests that the Board reverse the final rejections of the appealed claims.

Respectfully submitted,

Paul C. Gosnell
Registration No. 46,735

Attorney for Appellants

PCG14/ts

TAYLOR & AUST, P.C.
12029 E. Washington Street
Indianapolis, IN 46229
Telephone: 317-894-0801
Facsimile: 317-894-0803

Encs.: Return postcard

"EXPRESS MAIL" Mailing Number EV 620802268 US

Date of Deposit May 23, 2007.

I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 CFR 1.10 on the date indicated above and is addressed to the MS APPEAL BRIEF - PATENTS Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

Paul C. Gosnell, Reg. No. 46,735
(Typed Name of Person Mailing Paper or Fee)

(Signature of Person Mailing Paper or Fee)

IX. CLAIMS APPENDIX

1. A method of automatically assigning an internet protocol address to a device, comprising the steps of:

providing a network;

providing a computer communicatively coupled to said network;

providing a network adapter to communicatively couple said device to said network, said network providing communicative interconnection between said computer and said network adapter;

said computer performing the steps of:

generating an internet protocol address;

incorporating said internet protocol address in an address resolution protocol probe;

sending said address resolution protocol probe on said network; and

determining whether a response to said address resolution protocol probe indicates that said internet protocol address is in use;

wherein if said internet protocol address is not in use, then performing the step of assigning said internet protocol address to said network adapter via said network.

2. The method of claim 1, wherein if said internet protocol address is in use, then further comprising the step of repeating said generating step, said incorporating step, said sending step and said determining step.

3. The method of claim 2, further comprising the step of counting a number of times said generating step is performed.

4. The method of claim 3, comprising the step of comparing said number of times said generating step is performed to a predetermined number.

5. The method of claim 4, wherein said predetermined number is at least 30.

6. The method of claim 4, wherein if said number of times said generating step is performed exceeds said predetermined number then said computer does not automatically assign said network adapter an internet protocol address.

7. The method of claim 1, wherein prior to performing said generating step, said method comprising the steps of:

broadcasting a discovery packet on said network;

receiving a response from said network adapter; and

determining if said network adapter has a valid internet protocol address.

8. The method of claim 7, wherein prior to performing said generating step said method comprising the step of determining whether said network allows said computer to assign an internet protocol address to said network adapter.

9. The method of claim 1, wherein said device is a printer.

10. The method of claim 1, wherein said network adapter is a low-cost network adapter.

11. A method of automatically assigning an internet protocol address to a device, comprising the steps of:

providing a network;

providing a computer communicatively coupled to said network;

providing a low-cost network adapter to communicatively couple said device to said network, said network providing communicative interconnection between said computer and said low-cost network adapter;

said computer performing the steps of:

broadcasting a discovery packet on said network;

receiving a response from said low-cost network adapter;

determining if said low-cost network adapter has a valid internet protocol address;

wherein if said low-cost network adapter does not have a valid internet protocol address, then said computer performing the steps of:

generating an internet protocol address;

incorporating said internet protocol address in an address resolution protocol probe;

sending said address resolution protocol probe on said network; and

determining whether a response to said address resolution protocol probe indicates that said internet protocol address is in use;

wherein if said internet protocol address is not in use, then performing the step of assigning said internet protocol address to said low-cost network adapter via said network.

12. The method of claim 11, wherein if said internet protocol address is in use, then further comprising the step of repeating said generating step, said incorporating step, said sending step and said determining step.

13. The method of claim 12, further comprising the step of counting a number of times said generating step is performed.

14. The method of claim 13, comprising the step of comparing said number of times said generating step is performed to a predetermined number.

15. The method of claim 14, wherein said predetermined number is at least 30.

16. The method of claim 14, wherein if said number of times said generating step is performed exceeds said predetermined number then said computer does not automatically assign said low-cost network adapter an internet protocol address.

17. A network based imaging system, comprising:

a network;

a computer communicatively coupled to said network;

an imaging device; and

a network adapter communicatively coupling said imaging device to said network, said network providing communicative interconnection between said computer and said network adapter;

wherein said computer executes instructions which generate an internet protocol address, incorporate said internet protocol address into an address resolution protocol probe, send said address resolution protocol probe on said network, utilize a response to said address resolution protocol probe to determine if said internet protocol address is in use and if said internet protocol address is not in use, then assign said internet protocol address to said network adapter via said network.

18. The system of claim 17, wherein if said internet protocol address is in use then said computer repeats said instructions.

19. The system of claim 18, wherein said computer counts a number of times said instructions are executed.

20. The system of claim 19, wherein said computer compares said number of times said instructions are executed to a predetermined number.

21. The system of claim 20, wherein said predetermined number is at least 30.

22. The system of claim 20, wherein if said number of times said instructions are executed exceeds said predetermined number then said computer does not automatically assign said network adapter an internet protocol address.

23. The system of claim 17, wherein prior to performing said instructions said computer executes preliminary instructions which broadcast a discovery packet on said

network, receive a response from said network adapter and determine if said network adapter has a valid internet protocol address.

24. The system of claim 23, wherein said preliminary instructions further determine whether said network allows said computer to assign an internet protocol address to said network adapter.

25. The system of claim 17, wherein said network adapter is a low-cost network adapter.

X. EVIDENCE APPENDIX

Included herein, and listed below, is a copy of each reference upon which the Examiner relied in rejecting one or more of the claims of the present application.

Exhibit:

A. U.S. Patent No. 6,810,420 B1 (Buse).

B. Cheshire, S., Current Meeting Report, Cheshire, et al., 03/99 (Cheshire)

C. U.S. Patent No. 6,061,739 (Reed).

D. U.S. Patent No. 6,115,545 (Mellquist).

E. Troll, Request for Comments: 2563, May 1999, Troll R. (Troll)

XI. RELATED PROCEEDINGS APPENDIX

(No Entries)

US006810420B1

(12) **United States Patent**
Buse et al.

(10) **Patent No.: US 6,810,420 B1**
(45) **Date of Patent: Oct. 26, 2004**

(54) **ALLOCATION OF IP ADDRESS BY PROXY TO DEVICE IN A LOCAL AREA NETWORK**

(75) Inventors: **Christopher J Buse**, Watford (GB); **Andrew P White**, St Albans (GB); **David Kirby**, Hemel Hempstead (GB); **Robert Allsworth**, Abingdon (GB); **David E Bill**, St Albans (GB)

(73) Assignee: **3Com Corporation**, Santa Clara, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/494,401**

(22) Filed: **Jan. 31, 2000**

(30) **Foreign Application Priority Data**

Nov. 3, 1999 (GB) 9925897

(51) **Int. Cl.**[7] **G06F 15/173**
(52) **U.S. Cl.** **709/224**; 709/220
(58) **Field of Search** 709/222–224, 709/220

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,680,583 A		7/1987	Grover	
4,773,005 A		9/1988	Sullivan	
5,029,209 A		7/1991	Strong, Jr. et al.	
5,894,479 A	*	4/1999	Mohammed	370/401
5,978,373 A	*	11/1999	Hoff et al.	370/392
5,991,806 A	*	11/1999	McHann, Jr.	709/224
5,996,010 A	*	11/1999	Leong et al.	709/223
6,003,077 A	*	12/1999	Bawden et al.	709/223
6,006,019 A	*	12/1999	Takei	709/224
6,052,727 A	*	4/2000	Kamalanathan	709/224
6,070,187 A	*	5/2000	Subramaniam et al.	709/220
6,101,499 A	*	8/2000	Ford et al.	707/10
6,101,528 A	*	8/2000	Butt	709/203
6,157,956 A	*	12/2000	Jensen et al.	709/246
6,236,983 B1	*	5/2001	Hofmann et al.	706/47
6,282,575 B1	*	8/2001	Lin et al.	709/244
6,300,863 B1	*	10/2001	Cotichini et al.	340/5.8
6,360,260 B1	*	3/2002	Compliment et al.	709/224
6,490,617 B1	*	12/2002	Hemphill et al.	709/223
6,496,859 B2	*	12/2002	Roy et al.	709/223

OTHER PUBLICATIONS

Comer, "Internetworking with TCP/IP", vol. 1, 3rd Edition, 1995, pp. (368–369 and cover page) (4).*

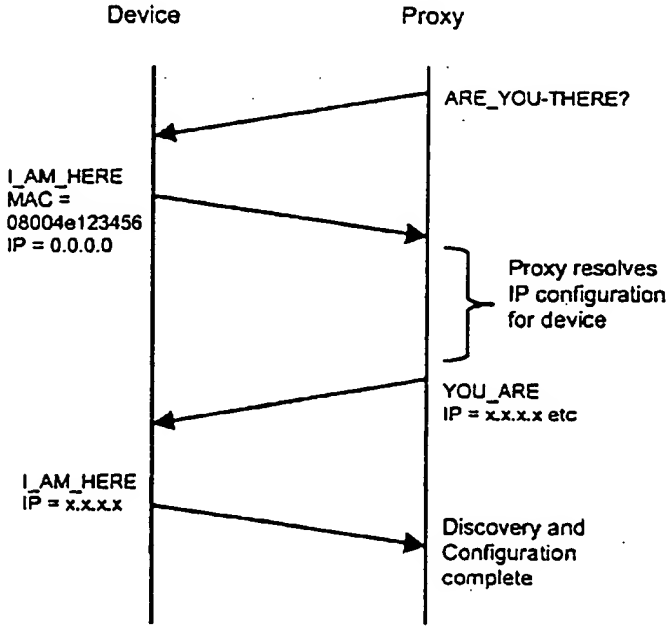
* cited by examiner

Primary Examiner—Bunjob Jaroenchonwanit
(74) *Attorney, Agent, or Firm*—Nixon & Vanderhye PC

(57) **ABSTRACT**

A device is discovered on a network by means of a discovery protocol operated by a proxy and a protocol address for the device is obtained if required by the proxy, which requests a protocol address in accordance with a dynamic host communication protocol, then in the absence of response to that request attempts automatic private addressing and finally if necessary allows manual entry of a protocol address for the device.

1 Claim, 3 Drawing Sheets



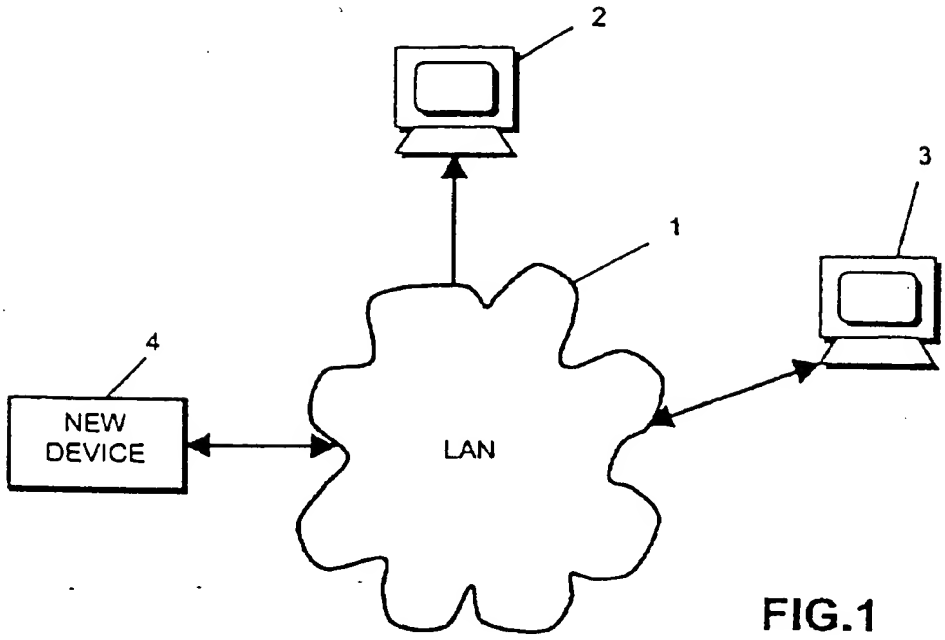

2
3
1
4
NEW DEVICE
LAN

FIG.1

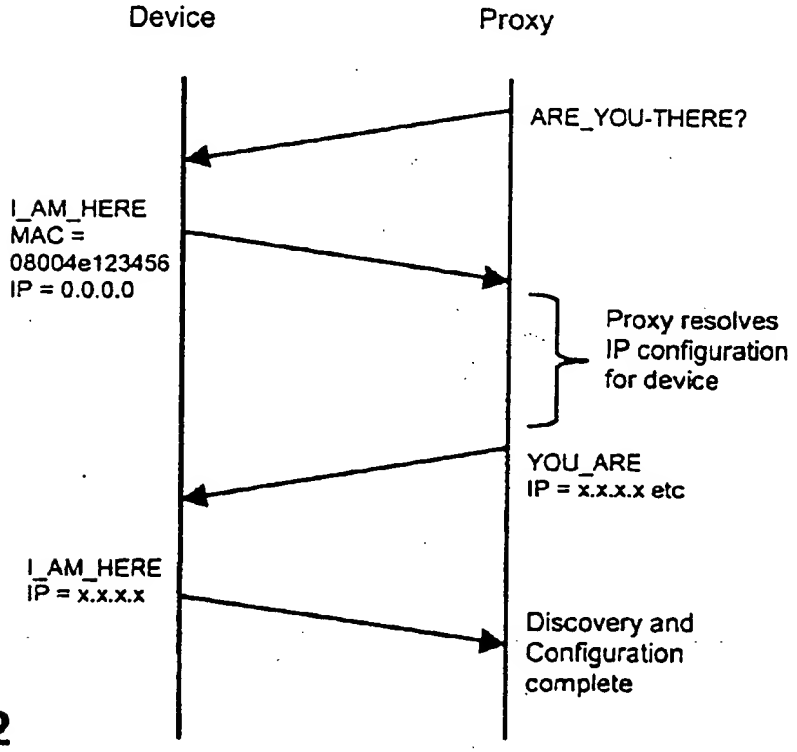

Device
Proxy
ARE_YOU-THERE?
I_AM_HERE
MAC =
08004e123456
IP = 0.0.0.0
Proxy resolves
IP configuration
for device
YOU_ARE
IP = x.x.x.x etc
I_AM_HERE
IP = x.x.x.x
Discovery and
Configuration
complete

FIG.2

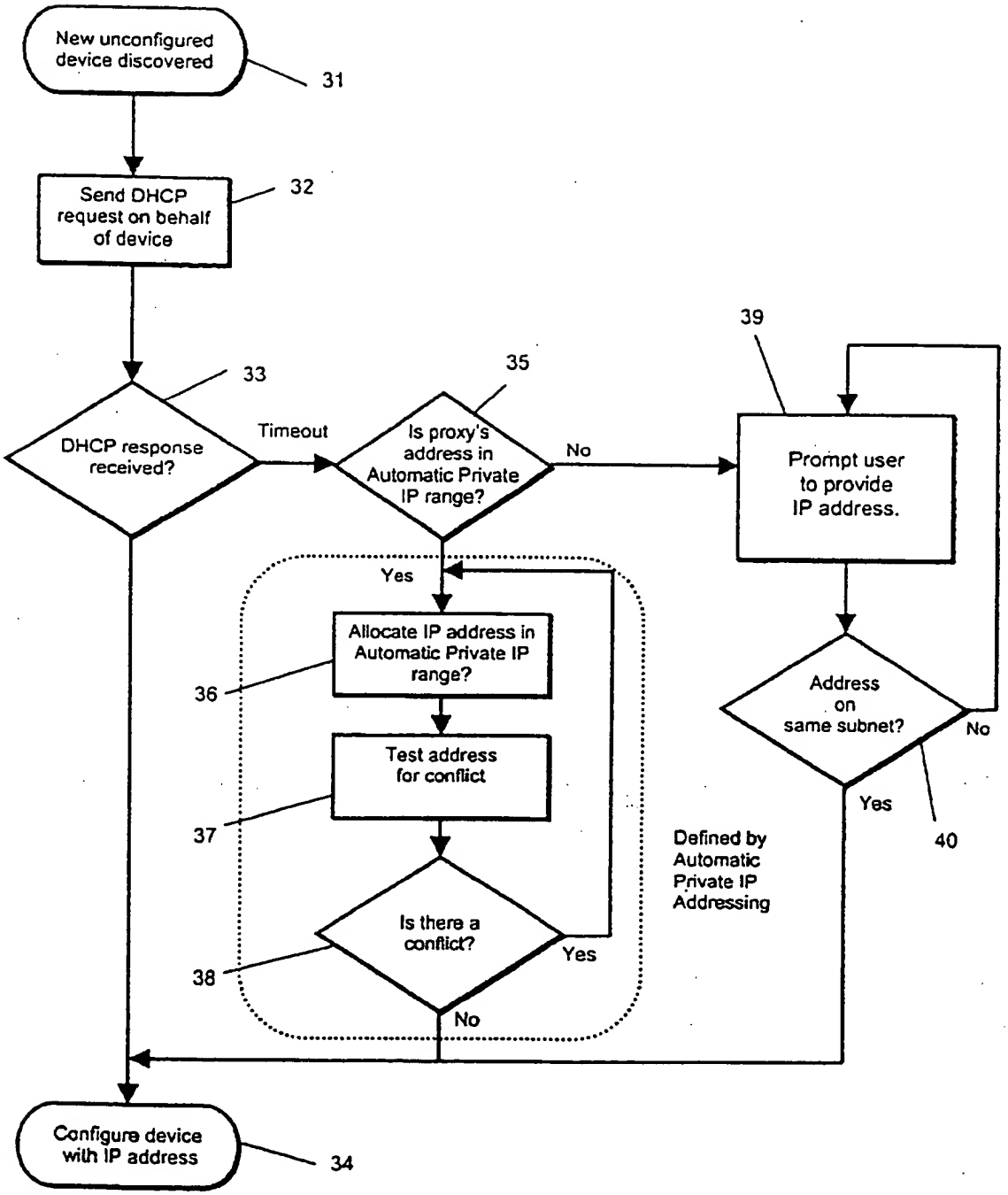

New unconfigured device discovered
31
Send DHCP request on behalf of device
32
DHCP response received?
33
Timeout
Is proxy's address in Automatic Private IP range?
35
No
Prompt user to provide IP address.
39
Yes
Allocate IP address in Automatic Private IP range?
36
Test address for conflict
37
Is there a conflict?
38
Yes
No
Address on same subnet?
No
Yes
40
Defined by Automatic Private IP Addressing
Configure device with IP address
34

FIG.3

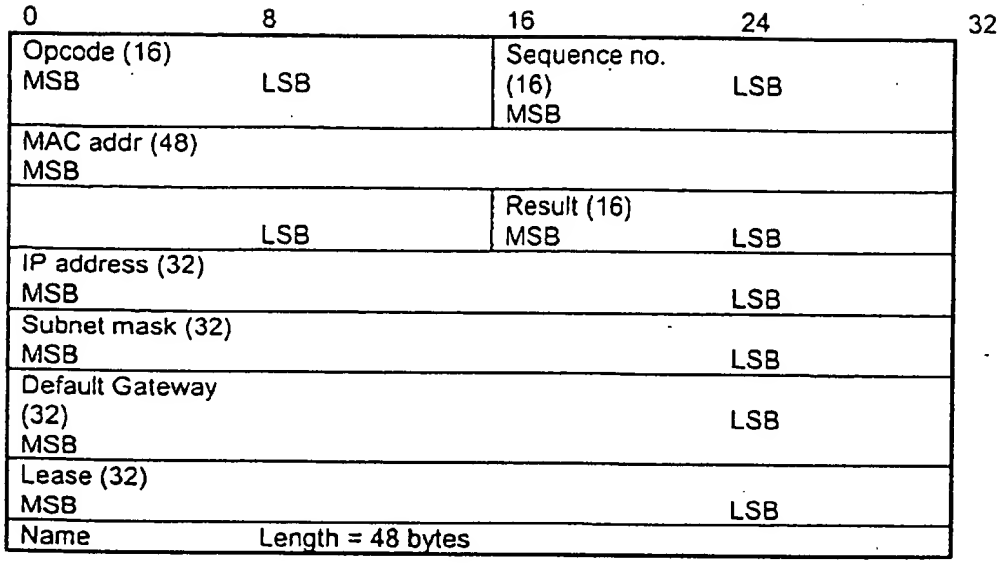


FIG.4

Field	Size(bytes)	Description
Op-code	2	1 = I_AM_HERE (sent by device, includes current units paramters) 2 = YOU_ARE (sent by proxy, includes parameters to use) 3 = ARE_YOU_THERE (sent by proxy, parameters ignored)
Sequence Number	2	Sequence Number. Will be set to a new value for each request, used to match requests with responses.
Mac Address	6	Unit hardware address or broadcast address for an initial ARE_YOU request frame
Result	2	0 = No error
IP Address	4	Unit configured IP address (0.0.0.0 if unconfigured)
Subnet Mask	4	Unit configured Subnet mask (0.0.0.0 if unconfigured)
Gateway	4	Unit configured default gateway (0.0.0.0 if unconfigured)
Lease	4	The length of time in seconds for device to use these IP address parameters. This is used by proxy to indiciate that the address is obtained via DHCP and has a finite lease time. 0xFFFFFFFF represents a non-expiring address.
Name	48	ASCII string with the description of the unit (e.g. 3Com OfficeConnect Dual Speed Switch 16). Padded with 0's, unless maximum length.

FIG.5

ALLOCATION OF IP ADDRESS BY PROXY TO DEVICE IN A LOCAL AREA NETWORK

FIELD OF THE INVENTION

This invention relates to packet-based data communication networks, particularly local area networks (LANs) and more particularly to the allocation of a protocol (IP) address to a device newly connected into the network

BACKGROUND OF THE INVENTION

When a new device is connected into an existing network, typically a local area network, it is desirable to determine whether such a device has a protocol address which is compatible with other devices on the network and to allocate the protocol address to the device if it does not already possess one. Another aspect of the process is the discovery of a device which does not have a protocol address on a network and to configure its protocol address in a convenient and compatible manner.

There are various schemes, such as dynamic host communication protocol (DHCP), and automatic private IP addressing, which can be used by a device to obtain a protocol address which is compatible with the network in which the device is to operate.

Ordinary network management discovery mechanisms generally rely on an existing configuration of devices for IP (internet protocol) and discover devices by performing a process known as ICMP Echo Request and Reply for all the IP addresses in a management station subnet. They presume that the user or network manager is knowledgeable.

Many devices have limited memory available for embedded software and are therefore not well adapted for direct participation in the programmed allocation of EP addresses.

SUMMARY OF THE INVENTION

One aspect of the present invention is a discovery scheme which can be operated by a proxy device such as a personal computer coupled to a local area network, and which facilitates the discovery of devices which may or may not be configured with an IP address.

Another aspect of the invention is a proxy scheme for the allocation of an IP address to a new unconfigured device discovered by the discovery mechanism.

Further objects and features will be apparent from the following description by way of example of a preferred scheme according to the invention

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates in simplified form a local area network and some devices connected thereto.

FIG. 2 schematically illustrates a discovery protocol.

FIG. 3 illustrates the operation of an IP address allocation scheme which may be performed by a proxy in a system according to FIG. 1.

FIG. 4 is a table showing the organisation and sizes of fields within control packets which may be employed in the present invention.

FIG. 5 is a table indicating the names and purposes of the fields within the control packets.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

FIG. 1 illustrates a local area network 1, comprising a multiplicity of interconnected devices. Coupled to the network is a user terminal 2 which may, particularly if it is operating under a 'Windows' (registered trade mark) operating scheme, host a DHCP (dynamic host communication protocol) and also an automatic private IP addressing scheme Reference numeral 3 denotes a PC coupled to the local area network and capable of operating the schemes described hereinafter.

Various allocation schemes may be used in networks of this general character. They are known as DHCP, automatic private IP addressing and the manual allocation of static addresses.

PCs that are configured to acquire an address by DHCP but fail to locate a DHCP server may be able to allocate themselves an address at pseudo random from a particular subnet. If the PC supports DHCP but not the self allocation feature it will attempt to operate with no IP address if DHCP fails to allocate an address. In this instance the PC will operate as a local machine but not participate in the network. If the PC can allocate its IP address it then performs an address resolution protocol on the address to check for any conflict and proceeds to use the allocated address.

FIG. 2 illustrates the discovery protocol which may be performed by the proxy in relation to a new device. FIG. 3 and 4 illustrate by way of example the control frames or packets that can be employed. The process employs three basic packets, distinguished by different operating codes ('Op-codes'). The proxy will periodically send an interrogation, identified herein as a frame having the 'ARE_YOU_THERE!' op-code Devices that see such an interrogation will respond with a reply, identified herein as a control frame having the 'I_AM_HERE' op-code. The device will return its MAC address (the third, 6-byte, field shown in FIGS. 4 and 5. Unconfigured devices will respond with an IP address field set to a convenially invalid value such as 0.0.0.0. Devices such as servers that have previously been configured will respond with an IP address field set to their IP address, completing the exchange. If desired or appropriate the device may return in the appropriate fields values for a subnet mask (set to an invalid value if the device is unconfigured), a default gateway (likewise set to an invalid value if the device is unconfigured) and a lease time which can be used by a proxy to indicate that the address is obtained by DHCP and is time-limited. A particular conventional value is employed to represent an infinite lease time.

A response from a device with an unconfigured IP address will initiate a process for resolving an IP address. That process is described with reference to FIG. 3 and in practice is mainly performed by software within the proxy 3.

When the proxy has resolved an IP address for the device it sends a declaratory message, identified herein as 'YOU_ARE' the message including the allocated IP address and preferably a subnet mask and other IP configuration parameters. Upon receipt of the 'YOU_ARE' message frame the device will configure itself with the supplied parameters and respond with a 'I_AM_HERE' message frame, the IP address field being set to the new IP address allocated to that device. If the allocation mechanism were through DHCP, the device will use that IP address for as long as its lease time allows. It will revert to 0.0.0.0 when the 'lease' expires. Periodically, the proxy may update (or validate) the least be interaction with the DHCP server to gain refreshed lease parameters. A special case of the lease is 'infinite' where the IP address is used for as long as the device is operational.

As will be apparent, an advantage of such a scheme is that the functionality associated with operating an IP address allocation scheme such as DHCP can be migrated from the

device to a proxy. In practice, this means that the device need only (for the purpose of address allocation) contain 'embedded' software sufficient to respond to the messages described previously. This requires minimal storage space in the device. New IP schemes require only an upgrade to an application which runs on the PC and will not normally require an upgrade of the devices. Furthermore, the scheme will operate over networks. It can be used to hide the MAC address from the user. The MAC address is used by the proxy to uniquely identify a device so that the device is not confused with other devices. If an IP address needs (as described later) to be entered by the user, the user is prompted to supply merely an IP address, there is no need to specify the MAC address because this is handled internally by the proxy.

FIG. 3 illustrates a scheme by which the IP address may be allocated to an unconfigured device newly discovered on the network. The stages shown in FIG. 3 are all performed by the proxy device 3 on behalf of the 'new' device 4.

Stage 31 indicates the discovery of a new unconfigured device. This corresponds to the receipt of the 'I_AM_HERE' message with the dummy IP address as described with reference to FIG. 2. The proxy 3 now sends a DHCP request on behalf of the device 4. If there is a DHCP server on the network, the proxy 3 will receive a DHCP response with an IP address for the device. The proxy can then send out a YOU_ARE frame configuring the device 4 with this IP address—stage 34. If the proxy does not receive a response to the DHCP request then it will timeout and move on to stage 35. If there is a time-out, there is a determination whether proxy's address is in the automatic private IP range. For the proxy to allocate IP addresses to devices using Automatic Private IP addressing, its own address must also be in this range. If its address is not in this range and the device is allocated an IP address from the auto IP range then the proxy will not be able to communicate with the device using IP as they will be in separate subnets. If it is in that range, an IP address may be automatically allocated, stage 36, the address may be tested for conflict with any existing addresses (stage 37). This may be done by means of an address resolution protocol (ARP) or an ICMP echo request. If there is a conflict, i.e. with an existing occupied address (stage 38) this sub process (stages 36–38) is repeated. Every time stage 36 is entered a new address is generated from the auto IP range for testing in stage 37. This cycle will continue until a free address is found. These three stages are all defined by an automatic private IP addressing scheme. If the proxy's address is not in an automatic private IP range, the user is prompted to provide an IP address, stage 39. On entry of the address the proxy checks whether the address is on the same subnet (stage 40). If necessary the manually entered IP address is conveyed to the device, stage 34.

What is claimed is:

1. A method of allocating an internet protocol address to a device connected to a packet-based communication network, comprising:

placing on the network an interrogation in the form of a first control frame from a proxy, said proxy being separate from said device;

receiving at the proxy a response from said device in the form of a second control frame which defines an invalid internet protocol address for said device;

in response to said invalid internet protocol address, sending from the proxy to a separate server a request for an internet protocol address for said device;

in response to the reception of said second control frame by said proxy, operating said proxy to test potential internet protocol addresses for conflict with existing internet protocol addresses, and obtaining said valid internet protocol address when conflict thereof with existing internet protocol addresses is absent by at least one of steps (a) to (c) as follows:

(a) by means of said request addressed according to a dynamic host communication protocol;

(b) automatic private internet protocol addressing; and

(c) manual entry of the internet protocol address, wherein said steps (a) to (c) are performed in the order (a),(b) and (c) until said valid internet protocol obtained; and

sending from the proxy to said device a third control frame which includes a valid internet protocol address allocated to said device.

* * * * *

- Current Meeting Report
- Slides

2.3.10 Networks in the Small - aka Home Networks (nits) bof

Current Meeting Report

Minutes: NITS BOF
15 March, 1999
Minneapolis IETF

CoChairs:
Stuart Cheshire, cheshire@apple.com
Peter Ford, peterf@microsoft.com
Mailing list: nits@merit.edu

After a short welcome and discussion of agenda we established a short list of presentations to discuss home networks. It appears that about 180-200 people attended the BOF, with a full room.

The list of presentations included:

1) Overall Goals of NITS BOF - Peter Ford

2) Home Networking Device and Service Discovery Requirements
- draft-miller-homedisc-req-00.txt - Brent Miller

3) Automatic IP address assignment for link local address w/IPv4
- draft-ietf-dhc-ipv4-autoconfig-03.txt - Stuart Cheshire/Ryan Troll

4) Multicast Discovery of DNS Services
- draft-manning-multicast-dns-01.txt - Bill Woodcock/Bill Manning

5) Service Location Protocol
- draft-ietf-svrloc-protocol-v2-12.txt - Erik Guttman

6) Simple Service Discovery Protocol
- draft-cai-ssdp-v1-00.txt - Yaron Goland

7) IPv6 Autoconfiguration and Neighbor Discovery - T. Narten

8) IP Address sharing - Stuart Cheshire

Goals of NITS BOF
Presented by Peter Ford and Stuart Cheshire

Goals articulated by Peter:

1) Discuss How Networks in the Small (e.g. Home Networks) can and should work
2) Determine if there is standards work to be done
3) If so, charter appropriate work
4) Non-goal - solve all problems by 5.30pm

Why should the IETF consider working on NITS now?

1) IP stack has sufficiently standardized and many vendors are looking at IP based devices that should have "plug and play behaviour".

2) IP currently fails the simplicity test - if one were to go to a CostCO and buy 2 PCs and an ethernet hub someone would still have to configure the IP addressing and other stack parameters.

Peter suggested that we should benchmark current IPv4 practices against those of Appletalk, IPX, and NetBIOS, including deployment of DHCP servers and bootstrapping thru DHCP to get DNS server addresses and domains configured in an end system. When one considers that to print many people then have to go edit their /etc/printcap and /etc/resolv.conf files we should turn red in embarrassment. With Appletalk and NetBIOS over NetBeui or IPX you can bring a system u p and print with minimal (close to zero) configuration.

Beyond simple addressing and DNS configuration we also need to worry about router, proxy and NAT configs that get more daunting every day.

Brent Miller from IBM Raleigh went through a brief description of the requirements his team from IBM developed for Home networks.

Basic assumptions include that there is a home LAN that is intermittently connected to the Internet and the LAN and LAN clients use standard IETF protocols. A computer that is introduced to this LAN should be able to enjoy basic connectivity to other similar systems and to services on that network. IN short, things should "just work".

Brent discussed a taxonomy of requirements from the draft including:
Autoconfiguration: IP address assignment to new devices, Service/Device location, and the use of User friendly names.

Question: Is this limited to single subnet single domain? What about multiple administrative domains? What if your teenage children want to run their own autonomous domains at home?
Answer: In order to make progress, should not require NITS solutions to scale beyond single subnet, single administrative domain.

Automatic IP address assignment for link local address w/IPv4
- draft-ietf-dhc-ipv4-autoconfig-03.txt - Stuart Cheshire/Ryan Troll

Stuart Cheshire presented a basic overview of IPv4 address self configuration as currently implemented by Apple and MS: The purpose is to support configuration of basic stacks without manual configuration. Stuart pointed out that both Apple and Microsoft are attempting to move towards complete use of IETF standard protocols in replacement of their legacy protocols (Appletalk and NetBeui).

Stuart started off his slide deck with a simple description of autonet:

1) Designed for small isolated LANs
- a home LAN
- a small office

2) Currently in Windows 98 and MAC OS 8.5

3) Described in an Internet Draft by Ryan Troll (Carnegie Mellon).

Stuart proceeded to describe the operation in Mac OS 8.5.

1) DHCP Discover
2) If no reply retry, 4, 8, 16 second intervals
4) If no DHCP server discovered then:

4a) Pick a random address on 169.254.*.* subnet (except first and last 256 addressed which are reserved for future use).

Send an ARP probe (ala DHCP conflict detection) to verify address is not already in use

If address in use, go back to 4a) - iterate at most 10 times and then fail stack initialization

Configure the interface with IP address, and start using interface

Every 5 minutes send single DHCP Discover to determine if a DHCP server has come online on the LAN, if so then proceed to normal DHCP client behavior upon DHCPOffer message reception.

Stuart then pointed out key points/features:

1) allows invisible use of IP - a user can go to the chooser using appletalk, but actually connects to it using autoconfigured TCP/IP

2) This is NOT a substitute with IPv6 - IPv6 is critical so that everyone can get globally unique public IP addresses

3) Stuart would like to also perform resource and service discovery using IP instead of Appletalk Name Binding Protocol (NBP).

There were questions about some of the DHCP specifics. Also several questions came up on who "owned" the 169.254/16 subnet, and Bill Manning offered that the IANA did at that is what WHOIS tells people who see this happening behind their firewalls!

MUDDS
Multicast Discovery of DNS Services
Presented by Bill Woodcock

Bill indicated that version 1 of the spec is in the Internet Draft directory but that he and Bill Manning are up to version 3 of the spec. He noted that people are looking for simple replacements for existing protocols such as Appletalk NBP and NetBIOS browsing.

There are several applications:

1) Bootstrapping DNS resolvers on clients when there is not a DHCP server

2) Discovery of unconfigured devices such as printers and routers

3) Let Apple deprecate the AppleTalk Chooser

Bill also noted that this mechanism could be used to provide a lightweight subset of SLP features. Steve Deering asked how this works and Bill briefly described how you can find SRV records on hosts that are listening to the multicast port. A question was asked about the hostname part of the SRV record if you were looking for "any printer on the LAN" and Bill noted that DNS supports the use of wildcards (*) in the names, something which raised several eyebrows in the audience. Erik Guttman asked about how do you get DNS response suppression by end nodes if a DNS server WAS present on a LAN to whit Bill described how one would first look for a DNS server via multicast and if a server was found you would only use unicast queries to that server. This raised several questions about the scoping of multicast requests and several references to the work in the multicast working groups on administratively scoped multicast.

Erik Guttman noted that you need to put in aggressive backoffs into the protocol to prevent swamping of the LAN, especially in wireless cases. He also noted that there were issues of trust models, which are contained in SLP.

Henry Sinnreich of MCI asked several questions about whether this mechanism could be extended to find public DNS servers. This was answered that it depends on careful configuration of multicast scopes.

Bill pointed out that the major advantage of using multicast for DNS discovery is that it uses pre-existing technologies: Multicast and DNS. The multicast usage employs a statically assigned address within the administrative local-scope and SRV records are used to describe network services such as DNS or printing. The DNS transaction works as normal except the initial query is performed within the multicast group rather than with a preconfigured DNS server.

SLP Overview by Erik Guttman

Erik gave a brief overview of how SLP works and the current standards status. he mentioned several small last minute additions to cover the cases where people are using directories such as LDAP and what a minimal subset of SLP would be. He also described some future work in the SLP group that would address issues that some parties have had with the use of SLP in environments with LDAP servers:

1) an LDAP server could emit DA Adverts allowing LDAP clients to use LDAP directly instead of using SLP front ending a DS.

2) Erik described how JINI and non-JINI services are discovered using SLP. Erik then compared SLP with multicast DNS:

Queries in SLP are of the form: "servers that ..." such as what are the printers that all have 721 dpi support.

SLP is carefully crafted to work in networks from small to large

Steve Deering proceeded to ask if any consideration was made in SLP to take advantage of many multicast addresses so that the multicast IP filter could filter out many questions irrelevant to the server instead of waking all servers with all SLP queries. Erik said this was in the early versions of the protocol, but that it was pulled out of SLPv2 due to no apparent interest.

Charlie Perkins noted that SLP was designed against a set of requirements that looked surprisingly similar to the requirements that Brent Miller presented at the beginning of the session.

Simple Service Discovery presented by Yaron Goland

Yaron discussed Microsoft's investigations into the subset of home networking including service discovery.

The problem statement is how systems can discover each other even if there is no network administrator or directory support. A subset of this problem is to discover directory support if present on the network.

The target environment anticipated is:

1) an IP network with multicast support
2) limited memory and storage - aggregate of 1M of memory or less
3) HTTP and XML support expected to be common for supporting device to device communication. Yaron noted that investigation of embedded web systems web sites indicate that web servers can be small (< 70Kbytes of code).
4) Example devices include: thermostats, security systems, CD players, printers, scanners, etc.

Solution Parameters include:

1) multicast IP is used to enable discovery in the absence of directory.
2) HTTP/XML based - to re-use stacks already present in the devices (small web servers that support UI, mgmt, etc.)
3) no parameters - services are identified with URIs, any more powerful discovery or parameter negotiation is done in a "higher" protocol or a service specific protocol (e.g. IPP).

Yaron noted the draft on SSDP in the Interdraft directory and also noted there were several items TBD. A proposed implementation is:

1) use HTTP over multicast UDP
2) Declaration based discivery using OPTIONS method and If header
3) annnouncement based discovery using ANNOUNCE method and resource-state header. This would allow new services to introduce themselves on the fly.

There was good followup discussion on the main differences btw SSDP and SLP. SSDP does not support queries of the form "who are the XXX servers that are ...", only dealing with "who are XXX servers?"

IPv6 autoconfiguration and link local addressing - Tom Narten

The goal is to present an overview of what was done and standardized in IPv6 to support small networks

For autoconfiguration IPv6 provides for:

PNP out of the box experience including support for the isolated dentist's office (single LAN that is not Internet connected) with support that scales to large enterprise.

All addresses are "leased"

IPv6 has built in support for multiple addresses per interface.

IPv6 also has scoped addressing:

1) Link-Local for communication with immediate neighbor
2) Site-Local which is analogous to IPv4 net 10/8
3) global addresses that are globally unique

These are all documented in RFCs (question from audience).

Link Local addresses are such that every node assigns link local addresses to its interfaces, those addresses are only good for that link/subnetwork. Link local addresses have a well known prefix (fe80::/9). Each address has an interface identifier that is derived from the MAC address, which is globally unique. There is built in duplicate address detection. It works much like the IPv4 autoconfiguration with a much lower probability of collision - probably none.

IPv6 supports both DHCP and stateless address autoconfiguration where there is no server on the wire (usually the router). A router sends a router advertisement and the end system cons's up an address from a prefix contained in the router advertisement (RA) and the Interface ID derived from the MAC of the Interface. If the interfaces sees multiple RAs then it means the host can have multiple addresses. One changes an interfaces address by changing the prefix contained in the RA. Tom noted that IPv6 uses dynamic DNS to update the DNS when an interface's address changes.

Router advertisements contain a default router list. Each host picks this up and maintains a local list of default routers. The host keeps track of reachability to all neighbors, if necessary the host can send probes.

IP address sharing
Stuart Cheshire/Apple Computer, Inc.

Stuart began describing the current status of getting PacBell ADSL at up to T-1 data rates. To get a single IP address for a single host the service is $50/month. If you want more IP hosts then it costs >$100 per month.

Stuart infers that IPv4 addresses are becoming scarce.

People are finding an arbitrage around additional cost per address in the deployment of NATs. He notes this is not an end-all solution. Nats are: fragile, break the end to end model, breaks IPSEC, requires separate support on a per protocol basis such as ftp and any other protocol that buries ports inside PDUs. The result is that NATs hold state on a per connection basis and is a single point of failure.

How could you make hosts behind NATs work better if they did know they were behind a NAT? Stuart poses that there can be address sharing aware (AAA) hosts. You need to ensure that on a single LAN where you are sharing the same IP address across hosts that hosts use unique to the LAN ports. The

NAT would need to be able to demultiplex inbound packets to hosts based on which hosts on the LAN have what ports in use. To that end Stuart proposed an "extended ARP" that is based on <IP address, proto, port> instead of simply using the IP addr. This can used as a claim mechanism as well as used by the NAT box to deliver a packet to the right host.

Stuart wanted to make it clear this was a 1-2 year solution and that IPv6 should be the real answer in the long term when it is fully deployed.

Several members in the group noted similar ideas were being discussed in the NAT working group and perhaps this topic should be out of scope of NITS due to active work in the NAT WG.

Closing Discussion

There appears to be sufficient interest in the area of small networks to proceed with forming a work list. Stuart and Peter took the action to work out a charter with the ADs (Thomas Narten and Erik Nordmark).

In enumerating the topics to be discussed we collected:

1) multicast DNS: more spec work needed, name resolution for IPv4 and v6, service discovery. Many voiced concern over m'cast scoping of queries. This topic had a lg amount of support for wg activity.

2) service discovery? mcast DNS for this purpose vs. SLP or SSDP? Issues to be determined was the spectrum of scalability, would mcast DNS suffice? Some mentioned applicability as an issue to be resolved.

3) What about IPv6, perhaps we should skip v4 and move directly to v6.

4) Need to refine requirements doc to aid in resolution of item 2) above.

5) What about multiple LANs (e.g. wireless, 1394 and ethernet all in one house)? Most felt we needed to worry about this topic.

6) what about multiple administrative domains on the same LAN such as in apartment buildings?

7) security was absent for most of the presentations, what are the security models and there was some who shared that they believed that security in the home was VERY important.

8) Is mobility an issue?

9) what about intranet vs Internet as a connectivity model (some said the use of the word intranet should be grounds for banishment ...). This appeared in the context of should the addresses on Home LANs be private or public, and how should 2 homes interconnect if private addresses are used

The meeting drew to a close at 6pm.

Slides

None received.

US006061739A

United States Patent [19]

Reed et al.

[11] Patent Number: 6,061,739

[45] Date of Patent: May 9, 2000

[54] NETWORK ADDRESS ASSIGNMENT USING PHYSICAL ADDRESS RESOLUTION PROTOCOLS

[75] Inventors: Benjamin Clay Reed, San Jose; Steven R. Welch, Gilroy, both of Calif.

[73] Assignee: International Business Machines Corp., Armonk, N.Y.

[21] Appl. No.: 08/979,141

[22] Filed: Nov. 26, 1997

[51] Int. Cl.[7] H04L 12/00

[52] U.S. Cl. 709/245; 709/222; 709/225; 709/228; 370/447; 370/471

[58] Field of Search 395/200.75, 200.52; 370/447, 471; 709/9, 222, 245, 225, 228, 231

[56] References Cited

U.S. PATENT DOCUMENTS

4,551,721	11/1985	Kozlik	370/452
4,689,786	8/1987	Sidhu et al.	370/431
4,727,475	2/1988	Kiremidjian	710/104
4,773,005	9/1988	Sullivan	710/9
5,150,464	9/1992	Sidhu et al.	709/222
5,166,931	11/1992	Riddle	370/401
5,237,614	8/1993	Weiss	380/23
5,287,103	2/1994	Kasprzyk et al.	370/351
5,355,375	10/1994	Christensen	370/407
5,434,918	7/1995	Kung et al.	380/25
5,446,897	8/1995	Mathias et al.	395/200.52
5,465,330	11/1995	Komatsu et al.	709/222
5,524,052	6/1996	Augustine et al.	380/49
5,526,489	6/1996	Nilakantan et al.	709/228
5,535,336	7/1996	Smith et al.	709/225
5,550,984	8/1996	Gelb	709/245
5,617,540	4/1997	Civanlar et al.	709/221
5,668,952	9/1997	Slane	395/200.75
5,781,552	3/1995	Hashimoto	370/447
5,854,901	7/1996	Cole et al.	395/200.75

OTHER PUBLICATIONS

Manual, Bootstrap and Autoconfiguration (BOOTP, DHCP), Chapter 21, pp. 365–379, Mar. 1995.

R. Droms, Memorandum re:Dynamic Host Configuration Protocol, Bucknell University, Mar. 1997, (pp. 1–45).

Finlayson, et al., Memorandum re: A Reverse Address Resolution Protocol, Stanford University, Jun. 1984, (pp. 1–4).

Bill Croft et al., Memorandum re: Bootstrap Protocol (BOOTP), Sep. 1985, (pp.1–12).

David C. Plummer, paper entitled An Ethernet Address Resolution Protocol or Converting Network Protocol . . . Ethernet Hardware, Nov. 1982, (8pp).

Primary Examiner—Ellis B. Ramirez

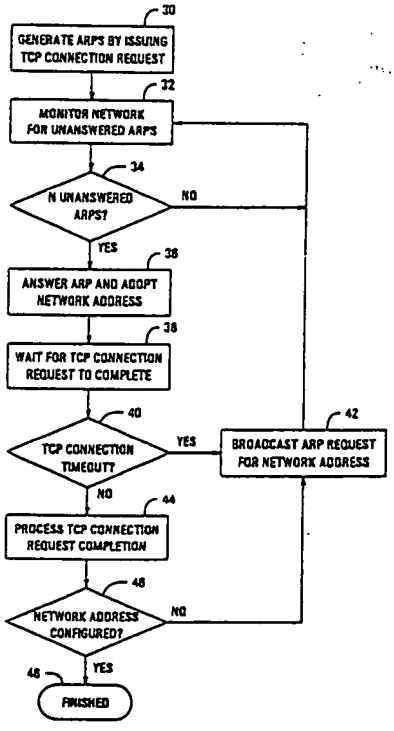
Assistant Examiner—Chuong Ho

Attorney, Agent, or Firm—Merchant & Gould P.C.

[57] ABSTRACT

A method, apparatus, and article of manufacture for initial network address configuration using physical address resolution protocol. A device attempts a connection to the network, which causes address resolution (ARP) packets to be generated. The device monitors communications on the network for a specified number of unanswered ARP packets. Thereafter, the device adopt the network address in the unanswered ARP packets and responds to the unanswered ARP packets with the its' physical address.

21 Claims, 2 Drawing Sheets



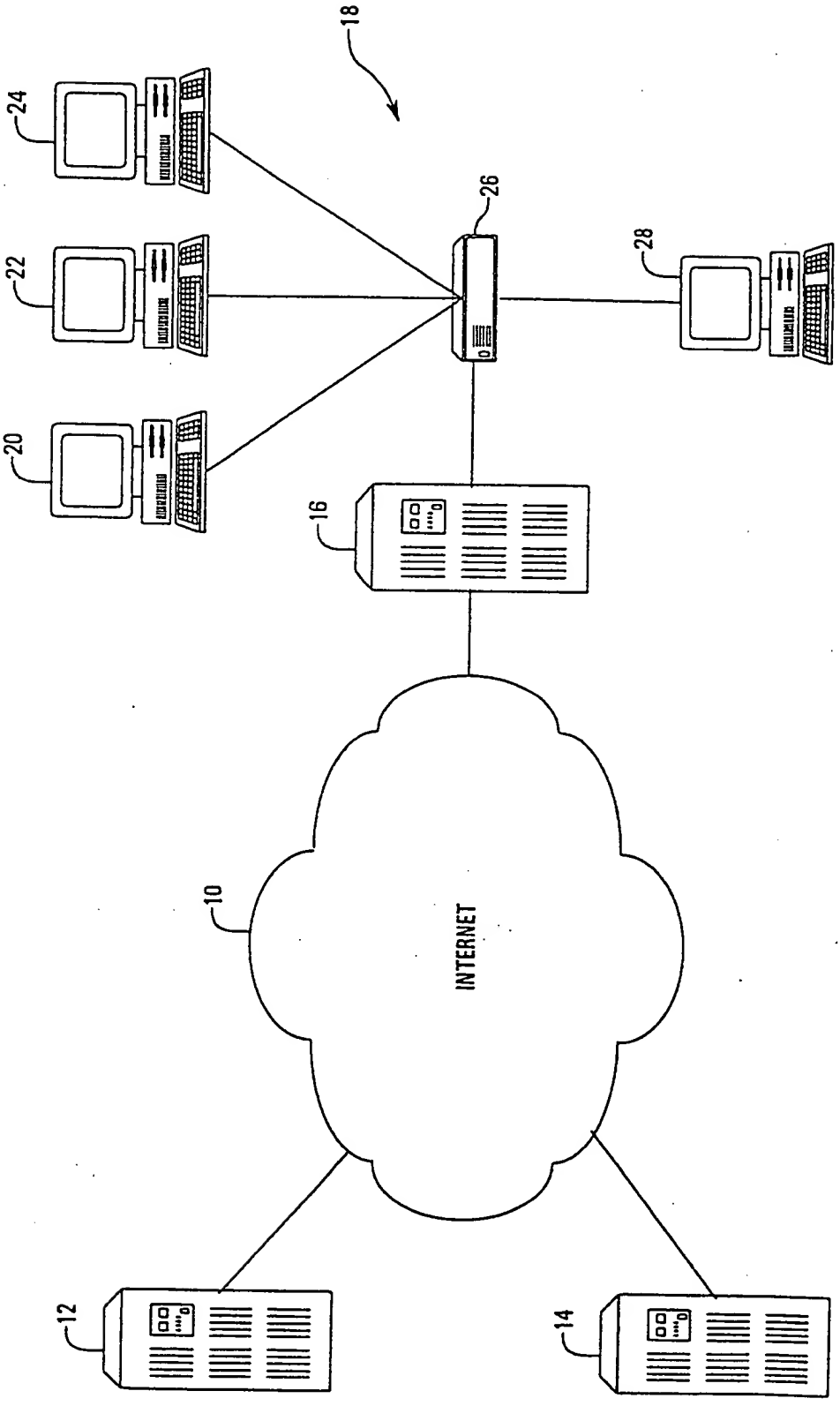
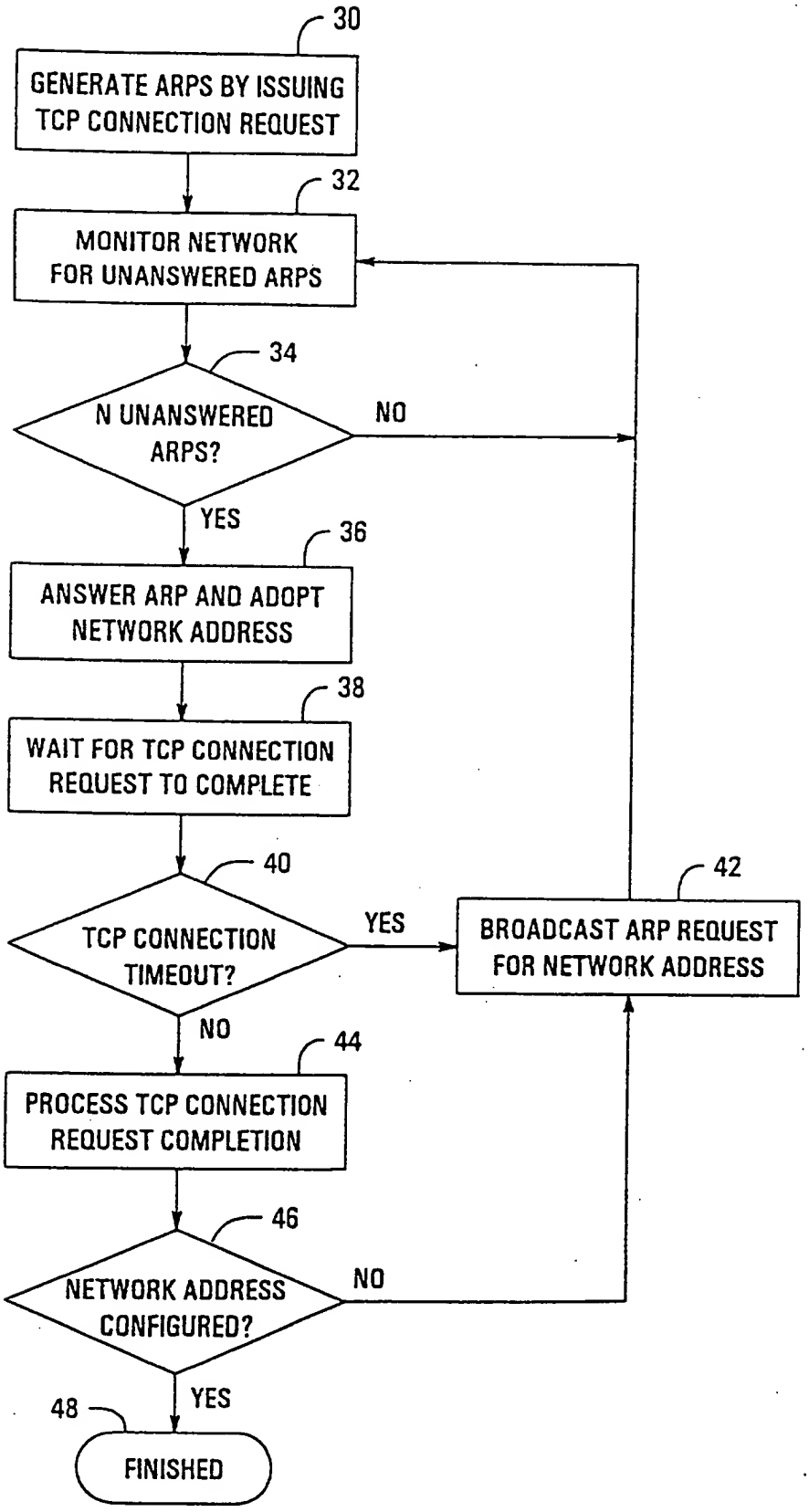


Fig. 1

Fig. 2


30
GENERATE ARPS BY ISSUING TCP CONNECTION REQUEST
32
MONITOR NETWORK FOR UNANSWERED ARPS
34
N UNANSWERED ARPS?
NO
YES
36
ANSWER ARP AND ADOPT NETWORK ADDRESS
38
WAIT FOR TCP CONNECTION REQUEST TO COMPLETE
40
TCP CONNECTION TIMEOUT?
YES
42
BROADCAST ARP REQUEST FOR NETWORK ADDRESS
NO
44
PROCESS TCP CONNECTION REQUEST COMPLETION
46
NETWORK ADDRESS CONFIGURED?
NO
YES
48
FINISHED

NETWORK ADDRESS ASSIGNMENT USING PHYSICAL ADDRESS RESOLUTION PROTOCOLS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to address resolution protocols for devices in communication networks, and more particularly, to systems that assign a network address using a physical address resolution protocol.

2. Description of Related Art

Transmission Control Protocol/Internet Protocol (TCP/IP) is a widely accepted international standard for describing communications on the Internet. In TCP/IP, network communications are divided into 4 layers, including application, transport, internet, and host-to-network layers.

An IP address is assigned to each host system or device operating within the Internet. The IP address includes a network address portion and a host address portion. The network address portion identifies a network within which the system resides, and the host address portion uniquely identifies the system in that network.

The combination of network address and host address is unique, so that no two systems have the same IP address. All IP addresses are 32 bits long and are usually written in dotted decimal notation, e.g., 255.255.255.255. These IP addresses are then used in the source address and destination address fields of IP packets.

In many cases, the network number in the IP address refers to a network on the Internet, such as a local area network (LAN), Internet Service Provider (ISP), intranet, etc. These networks may communicate with systems therein using other access protocols, such as Ethernet (IEEE Standard 802.3), dialup point-of-presence (POP), etc. These access protocols are typically broken down into the data link layer and physical layer, with the data link layer being further broken down into a media access control (MAC) layer and a logical link layer.

Systems such as personal computers, workstations, minicomputers, and mainframe computers attached to the networks each have a distinct lower level protocol identifier known as the physical network address or MAC address. Data forwarded to a destination system on the network under these lower level protocols contain the destination system MAC address, or other physical network address, as a destination address. Data forwarded from a source system on the network contain the source system MAC address, or other physical network address, as a source address. These systems thus communicate by encapsulating additional protocols within the lower layer protocols.

Systems routing data in an IP network rely on the network address portion of the IP address in a packet to find the network of the destination system. Once the network of the destination is located, the data is forwarded to that network and the host address portion is relied upon by the network to assign a MAC address for the destination system.

In order to communicate in this system, a network must first obtain its network address. Network numbers are assigned by the InterNIC (Internet Network Information Center) to avoid conflicts. Further, every system or device in a network must be assigned its own unique host address. Host addresses are usually assigned by network administrators and manually configured by a technician. Generally, the host addresses are associated with a hardware address for the system, such as an Ethernet address.

Every system or device that interfaces to a network usually has a unique hardware address. The hardware address is not location dependent and is globally unique. In general, the hardware address is set when a network interface is fabricated. Thus, when a network device (with network address x) wants to talk to another device (with network address y), the network device x must resolve network address y to a hardware address. The hardware address may be the address of the network interface card of device y or it may be the hardware address of the network interface of a gateway connecting device x to device y.

When a new system or device is initially attached to a network, that device must be given an IP address that maps to its unique hardware address. For systems or devices with consoles, the IP address can be keyed into a configuration screen. Systems or devices without consoles often have a serial port for attaching a terminal to perform this configuration operation. Alternatively, protocols exist that allow the new device to determine its IP address.

An Address Resolution Protocol (ARP), defined in RFC 826, provides a mechanism for determining IP addresses. For example, a first host having Ethernet address E1 broadcasts a packet onto an Ethernet LAN requesting a hardware address associated with an IP address. The broadcast is received by every system on the Ethernet LAN and each one checks its IP address for a match. When a second host at Ethernet address E2 identifies the IP address as its own, it responds to the broadcast with its Ethernet address E2. In this way, the first host learns that the requested IP address is associated with the second host having Ethernet address E2.

Various optimizations are possible to make ARP more efficient. To start with, once a system has run ARP, it may cache the result in case it needs to contact the same system shortly. Next time, it will find the mapping in its own cache, thus eliminating the need for a second broadcast.

Yet another optimization is to have every system broadcast its mapping when it boots. This broadcast is generally done in the form of an ARP looking for its own IP address. There should not be a response, but a side effect of the broadcast is to make any entry in every system's ARP cache. If a response does arrive, two systems have been assigned the same IP address, so the broadcasting system should inform the system manager and not complete its boot operation.

ARP solves the problem of finding out which Ethernet address corresponds to a given IP address. However, sometimes the reverse problem needs to be solved, i.e., given an Ethernet address, determining the corresponding IP address.

A reverse address resolution protocol (RARP), defined in RFC 903, has been developed to identify Ethernet addresses associated with IP addresses. As a result, the RARP allows a system without a configured IP address to obtain an IP address from a remote server. The system broadcasts a request and waits until a RARP server responds. In the request, the system must provide its physical network address (MAC address) to uniquely identify itself, allowing the server to map it into an IP address.

For example, the protocol allows a newly-booted system to broadcast its Ethernet address and request the corresponding IP address in response. The RARP server sees this request, looks up the Ethernet address in its configuration files, and sends back the corresponding IP address.

Using RARP is better than embedding an IP address in the memory image of the device, because it allows the same memory image to be used on all devices. If the IP address were buried inside the memory image, each device would need its own unique memory image.

The RARP protocol works fine, so long as a RARP server is available and is aware of the physical network address of the devices being added to the network. In order to find out the physical network address, all of the systems being added to the network must be passed through the RARP server, so that the address can be read from these systems, or a local technician must read the physical network address from the system and manually update the RARP server.

This process makes connecting a new device to a network difficult. Further, this process of physically reading the physical network address from the box is prone to human errors. Such addresses are typically very long (Ethernet MAC addresses are 48 bits long), and can be misread or typed in erroneously.

Another disadvantage of RARP is that it uses a destination address of all 1's (limited broadcasting) to reach the RARP server. However, such broadcasts are not forwarded by routers, so a RARP server is needed on each network.

To get around the problems associated with RARP, an alternative bootstrap protocol (BOOTP) has been developed, which is defined in RFCs 951, 1048, and 1084. Unlike RARP, BOOTP uses user datagram protocol (UDP) messages for connectionless transmission, which are forwarded over routers. It also provides a system with additional information, including the IP address of the file server holding the memory image, the IP address of the default router, and the subnet mask to use.

The Dynamic Host Configuration Protocol (DFCP) extends BOOTP in two ways. First, DHCP allows a system to acquire all the configuration information it needs in a single message. For example, in addition to an IP address, a DHCP message can contain a subnet mask. Second, DHCP provides a dynamic address allocation mechanism. Whenever a new device connects to the network, it contacts a DHCP server that maintains a set of IP addresses. The DHCP server chooses one of the addresses and then allocates the address to the device.

Notwithstanding these various techniques for assigning IP addresses to devices, there is still a need in the art for simplified mechanism for assigning IP addresses to devices. Moreover, there is a need in the art for techniques that can be performed by unskilled personnel. In addition, there is a need in the art for techniques that can be performed automatically without human intervention.

SUMMARY OF THE INVENTION

To minimize the limitations in the prior art described above, and to minimize other limitations that will become apparent upon reading and understanding the present specification, the present invention discloses a method, apparatus, and article of manufacture for initial network address configuration using physical address resolution protocol. A device attempts a connection to the network, which causes address resolution (ARP) packets to be generated. The device monitors communications on the network for a specified number of unanswered ARP packets. Thereafter, the device adopts the network address in the unanswered ARP packets and responds to the unanswered ARP packets with the its' physical address.

BRIEF DESCRIPTION OF THE DRAWINGS

Referring now to the drawings in which like reference numbers represent corresponding parts throughout:

FIG. 1 is a block diagram of an exemplary hardware environment of the preferred embodiment of the present invention, and more particularly, illustrates a typical configuration for the Internet; and

FIG. 2 is a flowchart that illustrates a state diagram of the logic performed by a device for network address assignment using physical address resolution protocols.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

In the description of the preferred embodiment, reference is made to the accompanying drawings which form a part hereof, and in which is shown by way of illustration the specific embodiment in which the invention may be practiced. It is to be understood that other embodiments may be utilized as structural changes may be made without departing from the scope of the present invention.

Overview

The present invention comprises a method for assigning a network address to a new device coupled to a network without any additional infrastructure or pre-existing knowledge of the hardware address of the device. After the device is attached to the network, it attempts to establish a connection on the network, which causes address resolution protocol (ARP) requests to be generated. The device monitors the communications on the network for unanswered ARP requests. When it sees N unanswered ARP requests (where N is a preset threshold) in a given length of time, the device adopts the requested network address and responds to the ARP with its hardware address.

Since the ARP is caused by a network connection attempt by the device itself, if the connection attempt does not succeed within a specified period of time, the ARP should not have been answered by the device. In this situation, the device broadcasts an ARP request for the network address it has adopted and starts monitoring for unanswered ARPs again. These steps are repeated until the device is configured with a correct IP address.

Hardware Environment

FIG. 1 is a block diagram of an exemplary hardware environment of the preferred embodiment of the present invention, and more particularly, illustrates a typical configuration for the Internet 10. A plurality of servers 12, 14, and 16 are connected to the Internet 10. An expanded view is provided of an Ethernet LAN 18 managed by one of the servers 16. One of the servers 16 manages an Ethernet LAN 18, wherein clients 20, 22, and 24 are connected to the server 16 via an Ethernet hub 26. In an exemplary embodiment, client 28 is a new device connected to the LAN 18 and must be configured with an IP address once it is connected to the Ethernet hub 26.

The servers 12, 14, and 16 and clients 20, 22, and 24 may comprise any device capable of being connected to a network. Generally these devices include, inter alia, a processor, random access memory (RAM), data storage devices, data communications devices, monitor, user input devices, etc. Those skilled in the art will recognize that any combination of the above components, or any number of different components, peripherals, and other devices, may be used without departing from the scope of the present invention.

The present invention is usually (although not necessarily) implemented by programming or logic embodied, executed, interpreted, and/or stored within the client 28. Thus, the present invention may be implemented

as a method, apparatus, or article of manufacture using standard programming and/or engineering techniques to produce software, firmware, hardware, or any combination thereof. The term "article of manufacture" as used herein is intended to encompass any programming or logic embodied, executed, interpreted, and/or stored within any device, carrier, or media.

Those skilled in the art will recognize that the exemplary environment illustrated in FIG. 1 is not intended to limit the present invention. Indeed, those skilled in the art will recognize that other alternative hardware environments may be used without departing from the scope of the present invention. For example, this invention is applicable to any type of network, configuration of devices, communications media, and is not limited to the networks, devices, and media shown in FIG. 1.

Flowchart

FIG. 2 is a flowchart that illustrates a state diagram of the logic performed by a device for network address assignment using physical address resolution protocols.

Initially, the device does not have a network address and Block 30 represents the device generating an ARP packet, for example, by issuing a TCP connection request.

Block 32 represents the device monitoring communications on the network for unanswered ARPs.

Block 34 is a decision block that represents the device determining whether N unanswered ARP requests have been observed in a first specified time interval, wherein both N and the first specified time interval are programmable values. If so, control transfers to Block 36; otherwise, control transfers to Block 32.

Block 36 represents the device answering the ARP requests with its own hardware address and adopting the network address specified in the ARP requests.

Block 38 represents the device waiting for completion of the previously-issued TCP connection request.

Block 40 is a decision block that represents the device determining whether the TCP connection request completed within a second specified time interval, wherein the second specified time interval is a programmable value. If not, control transfers to Block 42; otherwise, control transfers to Block 44.

Block 42 represents the device broadcasting an ARP request for the network address adopted by the device and associated with the device's hardware address, wherein the broadcast clears the ARP caches of the other devices on the network for this network address.

Block 44 represents the device processing the TCP connection by adopting the network address.

Block 46 is a decision block that represents the device determining whether the request configured the device's network address. If so, control transfers to Block 48, which represents the completion of the process; otherwise, control transfers to Block 42.

Conclusion

This concludes the description of the preferred embodiment of the invention. The following paragraphs describe some alternative methods of accomplishing the same objects.

The systems or devices being attached to the networks and being assigned IP addresses may include personal computers, workstations, minicomputers, mainframes, hubs, printers, or any other network-aware devices, etc. Indeed, any device that may be connected to a network would benefit from the present invention.

The present invention is not restricted to TCP/IP networks such as the Internet. Instead, the present invention would be applicable to intra-nets, LANs, WANs, SNA networks, or any other network.

In conclusion, the present invention discloses a method, apparatus, and article of manufacture for initial network address configuration using physical address resolution protocol. A device attempts a connection to the network, which causes address resolution (ARP) packets to be generated. The device monitors communications on the network for a specified number of unanswered ARP packets. Thereafter, the device adopt the network address in the unanswered ARP packets and responds to the unanswered ARP packets with its' physical address.

The foregoing description of the preferred embodiment of the invention has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. It is intended that the scope of the invention be limited not with this detailed description, but rather by the claims appended hereto.

What is claimed is:

1. A method of assigning a network address to a device coupled to the network using a physical address resolution protocol, comprising the steps of:
   (a) monitoring communications on the network to observe unanswered address solution protocol (ARP) packets therein;
   (b) adopting the network address in the unanswered ARP packets after a specified number of unanswered ARP packets have been observed, wherein the specified number of unanswered address solution protocol (ARP) packets is a programmable value; and
   (c) responding to the unanswered ARP packets with the device's physical address after the network address has been adopted.

2. The method of claim 1 above, wherein the monitoring step further comprises the step of monitoring communications on the network for the specified number of unanswered ARP packets in a given length of time.

3. The method of claim 2 above, wherein the given length of time is a programmable value.

4. The method of claim 1 above, further comprising the step of generating the ARP packets by attempting a connection to the network from the device.

5. The method of claim 4 above, further comprising the step of dropping the adopted network address when the attempt connection to the network from the device fails in a given length of time.

6. The method of claim 5 above, wherein the given length of time is a programmable value.

7. The method of claim 5 above, further comprising the step of repeating the generating, monitoring, adopting and responding steps when the adopted network address is dropped.

8. An apparatus for assigning a network address using a physical address resolution protocol, comprising:
   a device, coupled to a network, wherein the device monitors communications on the network to observe unanswered address solution protocol (ARP) packets therein, adopts the network address in the unanswered ARP packets after a specified number of unanswered

ARP packets have been observed, wherein the specified number of unanswered address solution protocol (ARP) packets is a programmable value, and responds to the unanswered ARP packets with the device's physical address after the network address has been adopted.

9. The apparatus of claim 8 above, wherein the device monitors communications on the network for the specified number of unanswered ARP packets in a given length of time.

10. The apparatus of claim 9 above, wherein the given length of time is a programmable value.

11. The apparatus of claim 8 above, wherein the device generates the ARP packets by attempting a connection to the network from the device.

12. The apparatus of claim 11 above, wherein the device drops the adopted network address when the attempt connection to the network from the new device fails in a given length of time.

13. The apparatus of claim 12 above, wherein the given length of time is a programmable value.

14. The apparatus of claim 12 above, wherein the device repeats the generating, monitoring, adopting and responding functions when the adopted network address is dropped.

15. An article of manufacture comprising a carrier embodying logic that causes a device connected to a network to perform a method for assigning a network address to the device using a physical address resolution protocol, the method comprising the steps of:

(a) monitoring communications on the network to observe unanswered address solution protocol (ARP) packets therein;

(b) adopting the network address in the unanswered ARP packets after a specified number of unanswered ARP packets have been observed, wherein the specified number of unanswered address solution protocol (ARP) packets is a programmable value; and

(c) responding to the unanswered ARP packets with the device's physical address after the network address has been adopted.

16. The article of manufacture of claim 15 above, wherein the monitoring step further comprises the step of monitoring communications on the network for the specified number of unanswered ARP packets in a given length of time.

17. The article of manufacture of claim 16 above, wherein the given length of time is a programmable value.

18. The article of manufacture of claim 15 above, further comprising the step of generating the ARP packets by attempting a connection to the network from the device.

19. The article of manufacture of claim 18 above, further comprising the step of dropping the adopted network address when the attempt connection to the network from the device fails in a given length of time.

20. The article of manufacture of claim 19 above, wherein the given length of time is a programmable value.

21. The article of manufacture of claim 19 above, further comprising the step of repeating the generating, monitoring, adopting and responding steps when the adopted network address is dropped.

* * * * *

US006115545A

United States Patent [19]

Mellquist

[11] Patent Number: 6,115,545

[45] Date of Patent: Sep. 5, 2000

[54] AUTOMATIC INTERNET PROTOCOL (IP) ADDRESS ALLOCATION AND ASSIGNMENT

[75] Inventor: Peter E. Mellquist, Auburn, Calif.

[73] Assignee: Hewlett-Packard Company, Palo Alto, Calif.

[21] Appl. No.: 08/891,624

[22] Filed: Jul. 9, 1997

[51] Int. Cl.[7] G06F 13/00

[52] U.S. Cl. 395/200.5; 395/200.51; 395/200.52; 395/200.53

[58] Field of Search 395/200.5, 200.51, 395/200.52, 200.53, 200.58, 200.59, 200.55, 200.56, 200.57

[56] References Cited

U.S. PATENT DOCUMENTS

5,557,748	9/1996	Norris	395/200.1
5,649,100	7/1997	Ertel et al.	395/200.1
5,724,510	3/1998	Arndt et al.	395/200.5
5,734,831	3/1998	Sanders	395/200.53
5,784,555	7/1998	Stone	395/200.5
5,819,042	10/1998	Hansen	395/200.52

OTHER PUBLICATIONS

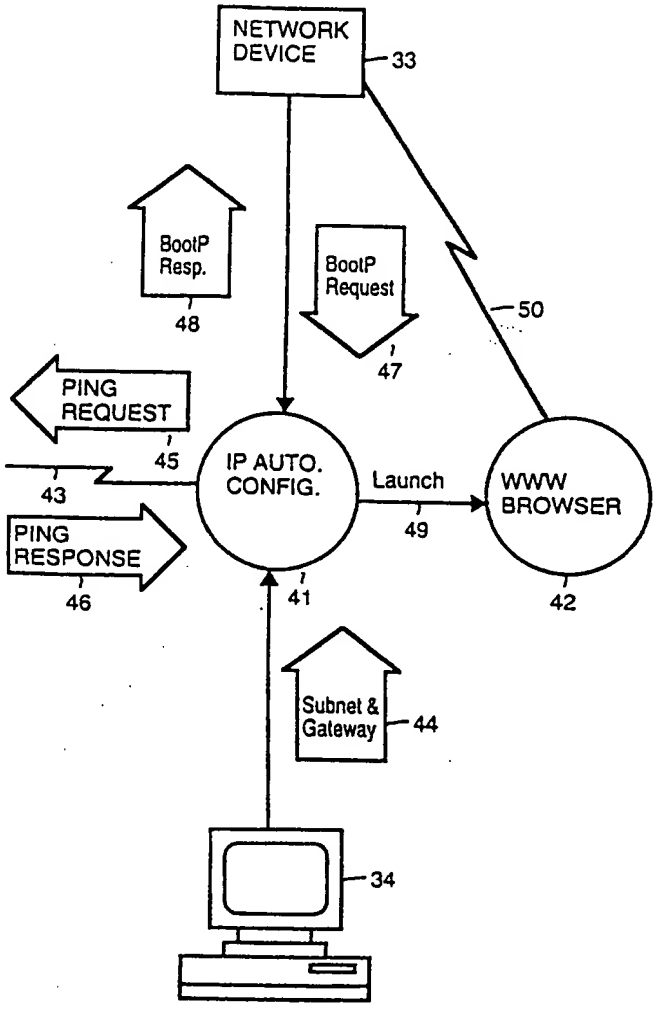
Douglas E. Comer, [Internetworking with TCP/IP:Principles, Protocols, and Architecture], vol. 1, Chapters 9 and 21, 3rd Edition, 1995.

Primary Examiner—Parshotam S. Lall
Assistant Examiner—Philip B. Tran

[57] ABSTRACT

A network device connected to a local network is configured using a module operating within a console connected to the local network. Once activated, the module obtains an unused network address. After obtaining the unused network address, the console waits for receipt of a request from the network device. Upon receipt of the request, the console forwards to the network device a response. The response includes the unused address along with subnet and gateway information for the console. The console then establishes a network connection to the network device and displays on a monitor for the console, an address value, a subnet mask value and a gateway value for the network device.

18 Claims, 4 Drawing Sheets



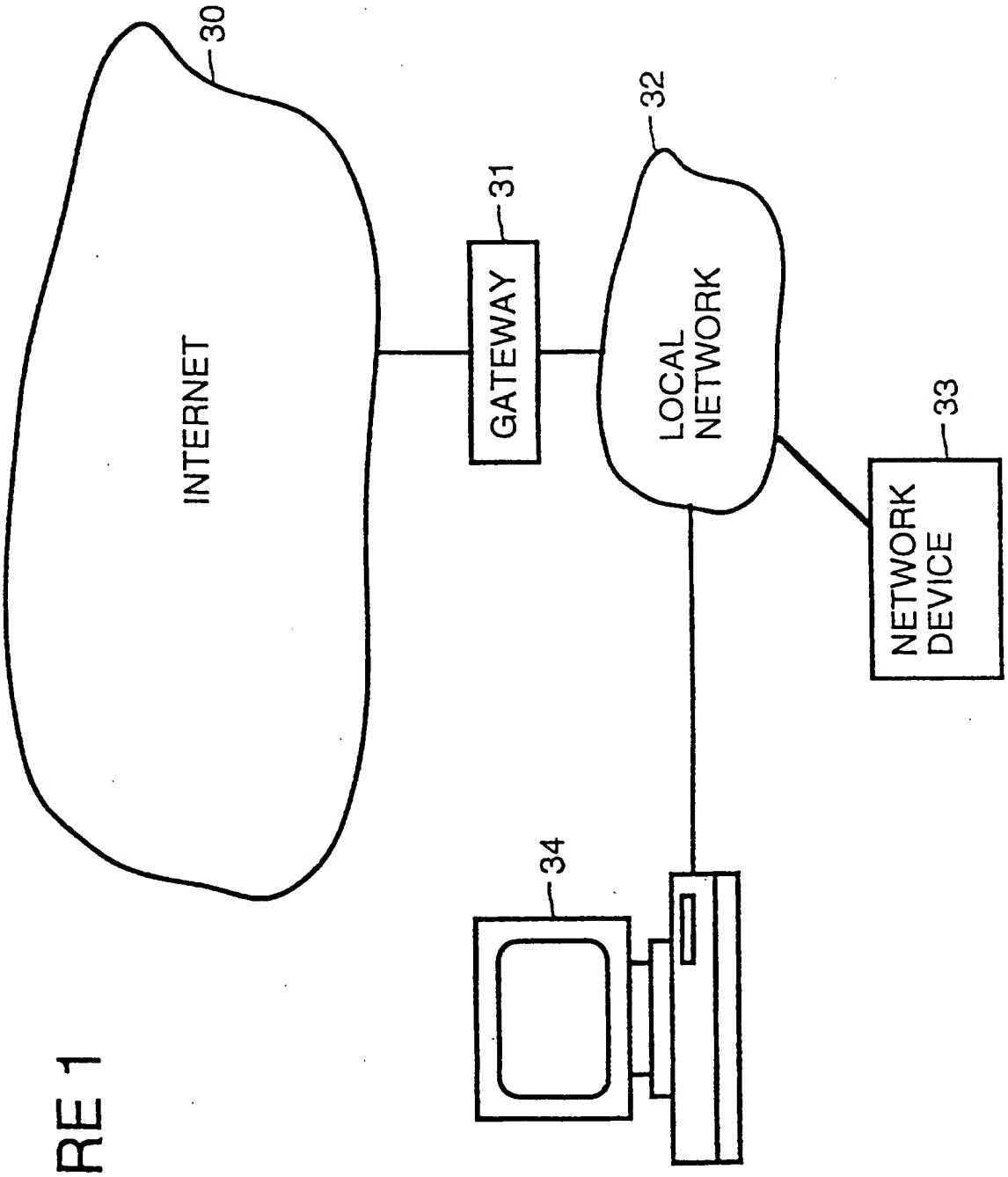

30
INTERNET
31
GATEWAY
32
LOCAL NETWORK
33
NETWORK DEVICE
34

FIGURE 1

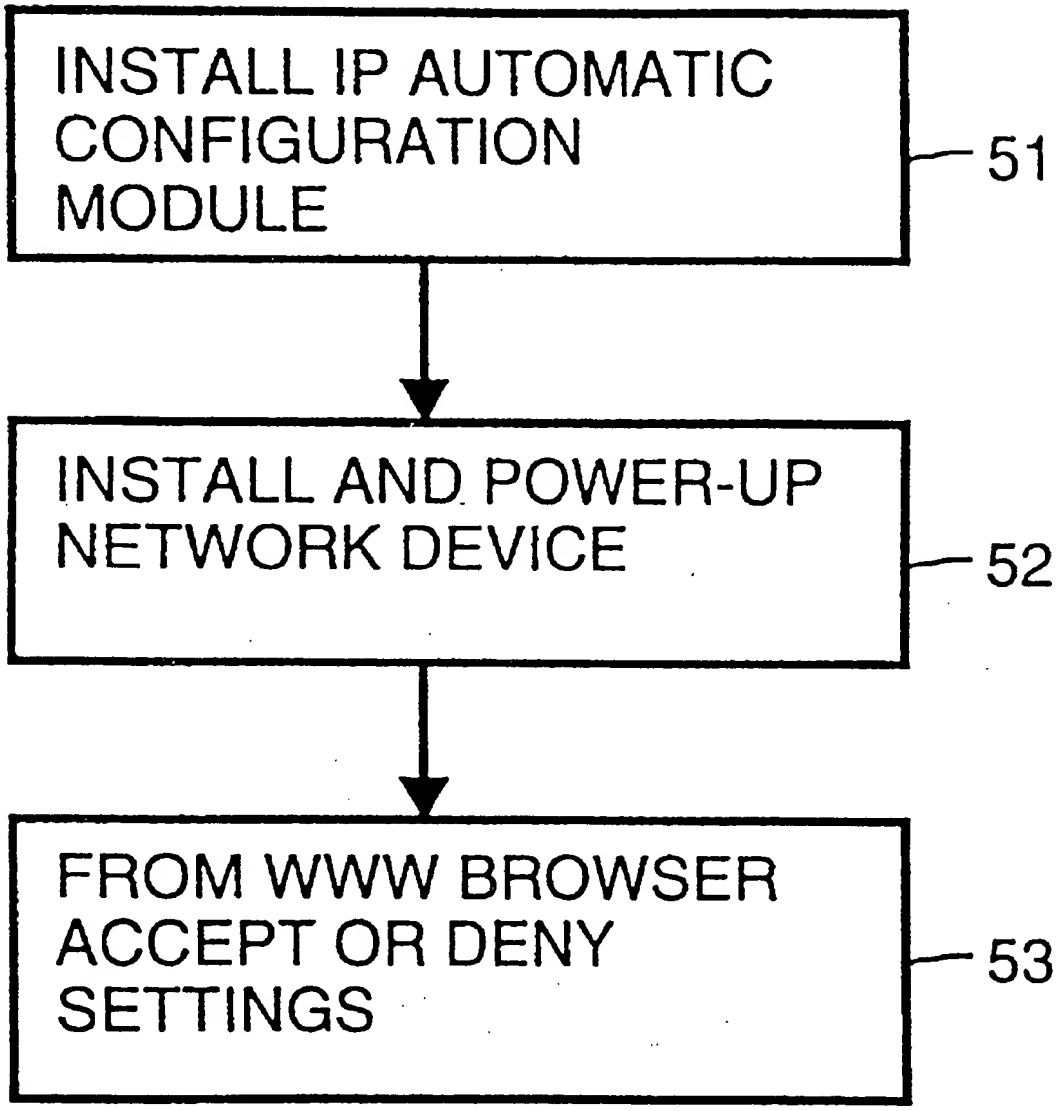

INSTALL IP AUTOMATIC CONFIGURATION MODULE
51
INSTALL AND POWER-UP NETWORK DEVICE
52
FROM WWW BROWSER ACCEPT OR DENY SETTINGS
53

FIGURE 2

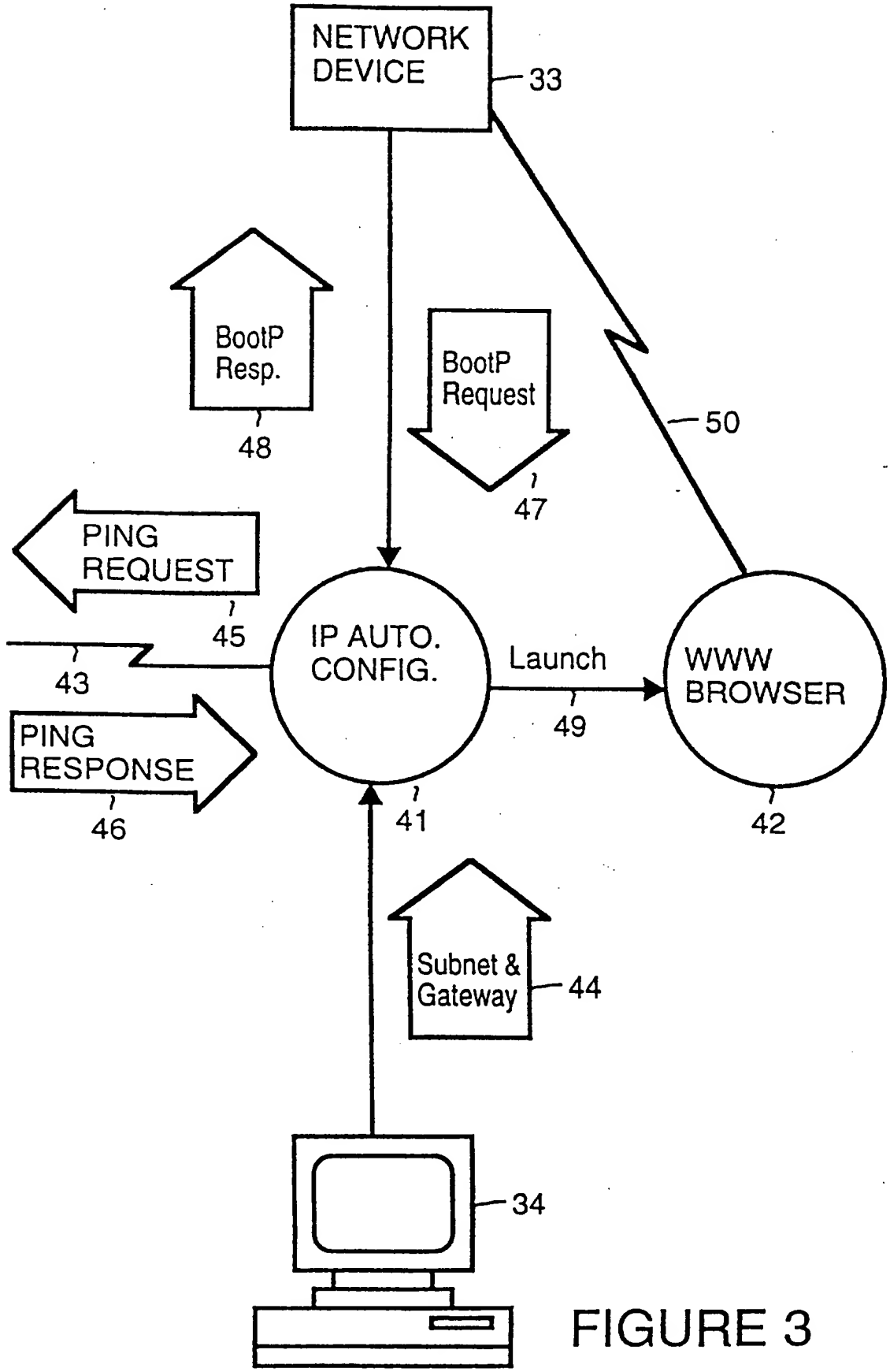

NETWORK DEVICE
33
BootP Resp.
48
BootP Request
47
50
PING REQUEST
45
43
IP AUTO. CONFIG.
Launch
49
WWW BROWSER
PING RESPONSE
46
41
42
Subnet & Gateway
44
34

FIGURE 3

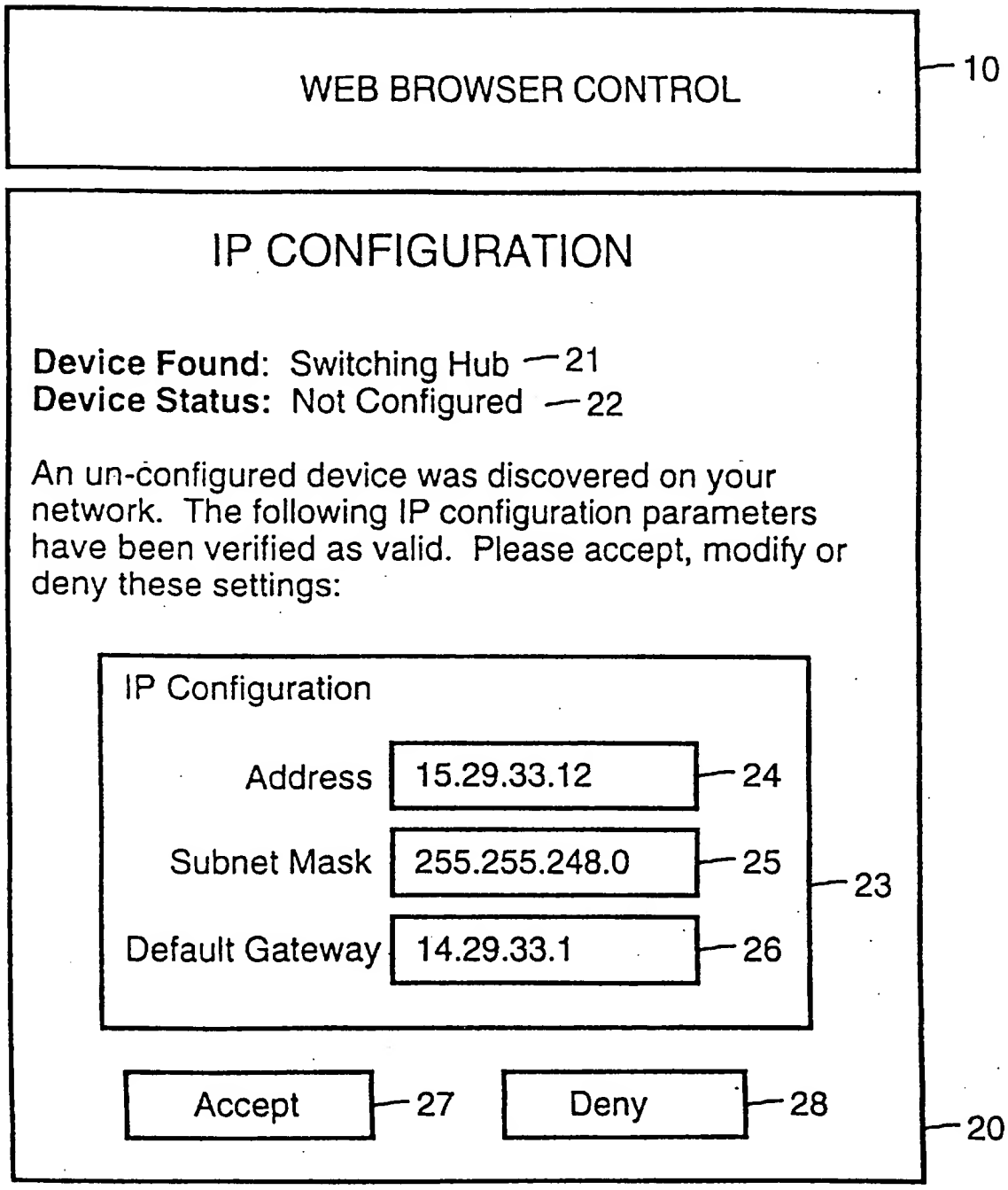

WEB BROWSER CONTROL
10
IP CONFIGURATION
Device Found: Switching Hub
21
Device Status: Not Configured
22
An un-configured device was discovered on your network. The following IP configuration parameters have been verified as valid. Please accept, modify or deny these settings:
IP Configuration
Address
15.29.33.12
24
Subnet Mask
255.255.248.0
25
Default Gateway
14.29.33.1
26
23
Accept
27
Deny
28
20

FIGURE 4

AUTOMATIC INTERNET PROTOCOL (IP) ADDRESS ALLOCATION AND ASSIGNMENT

BACKGROUND

The present invention concerns interconnected network devices and pertains particularly to automated internet protocol (IP) address allocation and assignment for the internet protocol.

The Transport Control Protocol/Internet Protocol (TCP/IP) has entered the main stream as the protocol of choice for network connectivity. TCP/IP commonly referred to as IP, has a number of benefits which attract networks and users including standardization, rich protocol and application support and the ability to route over Wide Area Networks (WANs). Although IP has many benefits, IP is difficult to configure and administer. This is especially the case in environments where a network novice may be present. Errors which may occur during IP installation can cause severe network problems and can be difficult to solve. The steps required to setup and configure an IP host are not intuitive.

Table 1 below sets out the parameters which are required for IP operation.

TABLE 1

Parameter	Where	Usage
Internet Protocol Address	Local Host	Required for all IP communication.
Sub-Network Mask	Local Host	Required for sub-net determination.

Table 2 below sets out optional configuration parameters for IP operation.

TABLE 2

Parameter	Where	Usage
Default Gateway	Local Host	Required only if multiple networks are present.
DNS Entry Definition	On DNS system	Required if friendly name usage is desired.
DNS Server Address	Local Host	Required on host for friendly name resolution.
WINS Server Address	On WINS system	Required for MS-Windows name resolution.
BOOTP Entry Definition	On BOOTP server	Required for automatic IP bootstrap.
DHCP Entry Definition	On DHCP server	Required for DHCP bootstrap.

There are a variety of ways to configure and setup an IP protocol stack. IP is flexible in that the protocol stack may be configured manually on the host, or automatically. The current solutions can be broken down into two main areas, basic and optional configuration.

For basic IP configuration, an IP address is selected and configured on the host entity. Selection of an IP address requires that the user knows a unique address which corresponds to the network where the host entity is to operate. IP addresses are typically managed by one central authority in order to guarantee uniqueness. Also, an IP sub-net mask must be selected and configured on the host entity. Selection of the IP sub-net mask is required such that the host's protocol stack can determine when an address is meant for the local sub-net verses when it should be passed to the default gateway.

For optional IP configuration, a default gateway must be selected and configured on the host entity. In order to allow communication across multiple networks, the default gateway must be configured. Also, a matching friendly name must be selected and configured on the Domain Name Server (DNS).

If a friendly name is to be associated with the configured IP address, the name needs to be configured on a Domain Name Server.

Also, a DNS address must be selected and configured on the host entity.

In order for the IP entity to communicate with other IP entities using friendly names, a Domain Name Server address needs to be configured. A Windows Internet Name Service (WINS) server must be selected and configured on the host entity if MS-Windows networking is to be used. An IP host which is running the Microsoft (MS) Windows operating system, available from Microsoft Corporation, having a business address at 16011 NE 36th Way, Redmond, Wash. 98073-9717, may optionally define a WINS server. WINS is a name resolution service that resolves Windows networking computer names to IP addresses (not unlike DNS) in a routed environment. A WINS server handles name registration, queries and releases.

An alternate method of defining IP configuration information is through the usage of a BOOTstrap Protocol (BOOTP). BOOTP allows clients to automatically receive all IP configuration information from a configured BOOTP server. This frees the user from having to configure individual entities but the BOOTP server itself needs to be configured. The BOOTP server serves IP information as well as vendor specific data to an entity which has broadcast a BOOTP request.

Also, configuration may be through a Dynamic Host Configuration Protocol (DHCP) Server. DHCP provides a framework for passing configuration information to hosts on a TCP/IP network. DHCP is based on BOOTP, but goes beyond by adding the capability of automatic allocation of reusable addresses and configuration options. Like BOOTP, the configuration for individual entities must be configured on the DHCP server. DHCP reuses IP addresses but does not address the issue of friendly names associated with these addresses and how their associations may change.

The current IP configuration schemes are workable but are not intuitive. A TCP/IP knowledgeable person is required to provide the basic configuration information. A person familiar with TCP/IP but not familiar with the network cannot properly configure IP since the currently available set of addresses, a sub-net mask and default gateway are required. An improperly configured IP stack can cause serious problems to existing networks. For many entities which do not have a keyboard and monitor this task is even more difficult. For these kind of systems, the basic IP parameters need to be configured either out-of-band or using an automatic bootstrap protocol.

One of the difficulties of IP configuration is that there are many ways to do it. Given an entity with a IP stack it can be configured using a local console, using an Out-Of-Band console, using an in-band console, using BOOTP or using DHCP.

Specifically, a local console can be used to locally configure an entity provided the entity has a keyboard and monitor. For entities which do not have a keyboard and console but do provide a form of Out-Of-Band interface, such as RS-232, IP parameters can be configured via an Out-Of-Band console.

For entities which already have an IP stack and would like to perform changes, an In-Band session may be utilized to perform IP configuration via an In-Band Console. Typically, this is in the form of a Telnet session.

For automatic bootstrapping, BOOTP may be utilized. Here the work required is on the BOOTP server which needs to have an entry for the machine booting. For Ethernet entities, the MAC address must be entered along with the matching BOOTP data. DHCP is similar to BOOTP only IP addresses are reused and conserved by DHCP.

In order to configure the IP stack, some basic information is required in order to do it safely. In order to define an IP address, a free address in the range of valid addresses must be selected. Addresses are usually administered by a person who allocates these addresses to entities who require them. It is important that duplicate addresses are not allowed since this can cause major trouble. Also, a sub-net mask is required for proper operation. The mask must be the same on all entities across the sub-net. In addition, if the network has more than one sub-net and a gateway exists, it must be configured such the host can take advantage of it. In order to do so, the IP address of the gateway must be configured. If automatic bootstrapping is to be utilized, the host must be configured to issue a BOOTP or DHCP request (many IP stacks may do this automatically if no local stack is configured). Configuring the BOOTP or DHCP server can be a significant amount of work.

SUMMARY OF THE INVENTION

In accordance with the preferred embodiment of the present invention, a network device connected to a local network is configured using a module operating within a console connected to the local network. Once activated, the module obtains an unused network address. This is done, for example, by the console sending a ping request via the local network and receiving a ping response via the local network. Addresses which do not respond may be deemed as "not currently used". Alternatively, unused network addresses may be obtained through other mechanisms such as: DHCP or BOOTP table reading, Simple Network Management Protocol (SNMP), Management Information Base (MIB) variable reading, and simple administrator input.

After obtaining the unused network address, the console waits for receipt of a request from the network device. Upon receipt of the request, the console forwards to the network device a response. The response includes the unused address along with subnet and gateway information for the console. For example, the request is a BOOTP request and the response is a BOOTP response. The network device is initially configured using the unused address and the subnet and gateway information within the response.

The console then establishes a network connection to the network device and displays on a monitor for the console, an address value, a subnet mask value and a gateway value for the network device. For example, this is done by launching a world wide web browser directed toward the newly issued address. The address value, the subnet mask value and the gateway value are displayed in a web page of the web browser. In the preferred embodiment, the module is a plug-in module of the world wide web browser.

Also in the preferred embodiment, the user via the console, is provided opportunity to accept, deny or modify the address value, the subnet mask value and the gateway value for the network device. This is done, for example, via the web page displayed by the world wide web browser.

The present invention facilitates simplified configuration of a network device in a TCP/IP environment. This is especially beneficial to novice network users. For example, using the present invention, a user no longer has to enter data via an RS-232 port or utilize a BOOTP/DHCP server device to get a network device up and running. A user, however, still has the flexibility to change an initially assigned address. Also the present invention is adapted to be compatible with the existing BOOTP mechanism and so can be used on a wide variety of devices. In addition the preferred embodiment of the present invention is adapted for use as a plug-in to world wide web browsers. Thus the present invention provides assistant for any novice in configuration of an IP entity.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a simplified block diagram of a connected networks.

FIG. 2 is a simplified flowchart which illustrates IP configuration of a network device in accordance with a preferred embodiment of the present invention.

FIG. 3 is a data flow diagram which illustrates IP configuration of a network device in accordance with a preferred embodiment of the present invention.

FIG. 4 shows a configuration Hyper Text Transfer Protocol (HTTP) "home page" used during IP configuration of a network device in accordance with a preferred embodiment of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT

FIG. 1 is a simplified block diagram of a local network 32. Local network 32 is, for example, one or a combination of local area networks. A local console 34 and a network device 33 are shown connected to, and may be considered a part of local network 32. Local network 32 is, for example, connected to internet 30 through a gateway 31. Local console 34 is, for example, a personal computer.

FIG. 2 illustrates a method for performing IP configuration of network device 33 in accordance with a preferred embodiment of the present invention. Data flow for the method is illustrated by FIG. 3.

In a step 51, shown in FIG. 2, an IP configuration module plug-in is installed into a World Wide Web (WWW) browser running within local console 34. For example, the WWW browser is a Netscape Navigator web browser available from Netscape Communications Corporation, or a Microsoft Internet Explorer web browser available from Microsoft Corporation, having a business address at 16011 NE 36th Way, Redmond, Wash. 98073-9717. Plug-ins are a standardized extension mechanism for WWW browsers. Plug-ins are typically used for decoding new media types but can be used for other purposes as well such as application integration with a WWW browser.

For example, FIG. 3 shows an IP configuration module 41 which functions as a plug-in within a WWW browser 42. Local console 34 can initially receive IP configuration module 41 from a floppy disk, a CD ROM, over internet 30 or from another entity on local network 32. Once activated, for example by a user of local console 34 or via a local Uniform Resource Locator (URL), IP configuration module operates in accordance with the algorithm set out in Table 3 below:

TABLE 3

```
Find a free IP address using Random ICMP ping, discovery,
   DNS, ARP or RARP
Open BOOTP Server Port
On BOOTP Request DO
   if BOOTP Request is from recognized MAC address range
THEN
      Issue BOOTP response using free IP address, local sub-
        net mask and local gateway
      Invoke browser using free address in URL
```

Once activated, IP configuration module 41 will find an address which is not in use. This is determined through a variety of techniques. For example, the address may be obtained using an Internet Control Message Protocol (ICMP) Ping. An ICMP Ping is a simple protocol used to determine connectivity and usage of an IP address. In addition, ICMP Ping may be used to determine round trip response times in a network protocol stack.

In addition, a Domain Name Server (DNS) address can be obtained. That is, using a DNS configuration table, addresses currently in use, or those about to be used can be determined. Also, addresses which will not be used can be determined and thereby be utilized as free addresses. Also an ARP cache reading can be performed.

For example, in FIG. 3, data flow for obtaining the address is illustrated by a ping request 45 and a ping response 46 make over datapath 43 to local network 32. Ping responses denote addresses being used.

In addition, IP configuration module 41 utilizes the sub-net mask and default gateway of local console 34 for configuring network device 33. This is illustrated by sub-net mask and gateway information 44 shown being transferred from local console 34 to IP configuration module 41.

Then IP configuration module 41 acts in place of a BOOTP server to accept and reply to a select set of BOOTP requests from devices having a recognized Media Access Control (MAC) address range. MAC addresses are used for level 2 addressing in the OSI 7 level model. A BOOTP request may contain a level 2 MAC address for an entity which requires a level 3 address. For example, using Ethernet protocol, a BOOTP request would contain an Ethernet MAC address. The corresponding BOOTP response would contain an issued IP address to that MAC address.

Once a valid BOOTP request has been received, IP configuration module 41 will issue a BOOTP response which includes the address which it has found to not be in use. The BOOTP response uses the sub-net mask and default gateway of local console 34. This is illustrated by BOOTP request 47 and BOOTP response 48 shown in FIG. 3 being transferred between network device 33 and IP configuration module 41.

Once BOOTP response 48 has been issued, IP configuration module 41 will invoke WWW browser 42 to point to the device address just issued. The Uniform Resource Location (URL) for network device 33 will include the newly issued address. The Uniform Resource Indicator (URI) will specify a configuration content page 20, shown in FIG. 4. Once network device 33 has accepted the address, the web server within network device 33 will accept requests from IP configuration module 41 and allow configuration content page 20 to be displayed within WWW browser 42.

In a step 52, shown in FIG. 2, network device 33 is installed in local network 32 and powered up. Once powered up, network device 33 will issue a broadcast BOOTP request (i.e., BOOTP request 47) which will be picked up by IP configuration module 41, as described above. IP configuration module 41 will issue BOOTP response 48 by which network device 33 will obtain the IP configuration parameters and proceed to initialize. WWW browser 42 is then launched, as illustrated by arrow 49. When launched, WWW browser 42 is pointed at network device 33. A Web Server running on network device 33 will respond to WWW browser 42 providing configuration information which WWW browser 42 displays on configuration content page 20 along with a web browser control panel 10.

In a step 53, shown in FIG. 2, configuration content page 20 is used to accept or deny the settings. That is, once network device 33 has been powered up, the operator can return to local console 34. Local console 34 will display configuration content page 20 with the configuration parameters for network device 33. This network connection is illustrated by network connection 50, shown in FIG. 3.

For example, as shown in FIG. 4, configuration content page 20, in a location 21, lists the type of device which is found. For example, configuration content page is a world wide web page. In a location 22, configuration content page 20 lists the status of the device found. In a location 24, within an IP configuration box 23, configuration content page 20 lists an IP configuration address assigned to network device 33 by BOOTP response 48. In a location 24, configuration content page 20 lists a subnet mask assigned to network device 33. In a location 26, configuration content page 20 lists a default gateway for network device 33.

A user may modify the IP configuration address in location 24, the subnet mask in location 25 and/or the default gateway in location 26. The user can deny the configuration by selecting a deny button 28. The user can accept the original or the modified configuration by selecting an accept button 27.

The foregoing discussion discloses and describes merely exemplary methods and embodiments of the present invention. As will be understood by those familiar with the art, the invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. Accordingly, the disclosure of the present invention is intended to be illustrative, but not limiting, of the scope of the invention, which is set forth in the following claims.

I claim:

1. A computer implemented method used in configuration of a network device connected to a local network comprising the following steps performed by a module operating within a console connected to the local network:

(a) in response to activation of the module, obtaining an unused network address;

(b) upon receipt by the console of a request from the network device, forwarding to the network device a response which includes the unused address obtained in step (a) along with subnet and gateway information for the console;

(c) establishing a network connection to the network device; and,

(d) displaying on a monitor for the console, an address value, a subnet mask value and a gateway value for the network device.

2. A computer implemented method as in claim 1 wherein step (a) comprises the following substeps:

(a.1) sending a ping request via the local network; and,

(a.2) receiving a ping response via the local network.

3. A computer implemented method as in claim 1 wherein in step (b) the request is a BOOTP request and the response is a BOOTP response.

4. A computer implemented method as in claim 1 additionally comprising the following step:

(e) providing opportunity for a user via the console to accept, deny or modify the address value, the subnet mask value and the gateway value for the network device.

5. A computer implemented method as in claim 1 wherein step (d) includes launching a world wide web browser by which the address value, the subnet mask value and the gateway value are displayed.

6. A computer implemented method as in claim 5 wherein the module is a plug-in module of the world wide web browser.

7. A method for configuring a network device connected to a local network comprising the following steps:

(a) from a console located within the local network, interrogating active network devices to determine an unused network address;

(b) upon the console receiving a request from the network device, forwarding, by the console to the network device, a response which includes the unused address determined in step (a) along with subnet and gateway information for the console; and,

(c) configuring of the network device using the unused address and the subnet and gateway information within the response forwarded in step (b).

8. A method as in claim 7 wherein step (a) comprises the following substeps:

(a.1) sending a ping request from the console out to the local network; and,

(a.2) receiving a ping response by the console from the local network.

9. A method as in claim 7 wherein in step (b) the request is a BOOTP request and the response is a BOOTP response.

10. A method as in claim 7 additionally comprising the following steps:

(d) establishing a network connection between the console and the network device; and,

(e) displaying on a monitor for the console, an address value, a subnet mask value and a gateway value for the network device.

11. A method as in claim 10 additionally comprising the following step:

(f) providing opportunity for a user via the console to accept, deny or modify the address value, the subnet mask value and the gateway value for the network device.

12. A method as in claim 10 wherein in step (e) the address value, the subnet mask value and the gateway value are displayed within a window of a world wide web net browser.

13. Storage media which stores a software module, the software module, when run on a console connected to a local network, performing a method used in configuration of a network device connected to the local network, the method comprising the following steps:

(a) in response to activation of the software module, obtaining an unused network address;

(b) upon receipt by the console of a request from the local network, forwarding to the local network, a response which includes the unused address found in step (a) along with subnet and gateway information for the console;

(c) establishing a network connection between the console and the network device; and,

(d) displaying on a monitor for the console, an address value, a subnet mask value and a gateway value for the network device.

14. Storage media as in claim 13 wherein step (a) comprises the following substeps:

(a.1) sending a ping request via the local network; and,

(a.2) receiving a ping response via the local network.

15. Storage media as in claim 13 wherein in step (b) the request is a BOOTP request and the response is a BOOTP response.

16. Storage media as in claim 13 wherein the method additionally comprises the following step:

(e) providing opportunity for a user via the console to accept, deny or modify the address value, the subnet mask value and the gateway value for the network device.

17. Storage media as in claim 13 wherein step (d) includes launching a world wide web browser by which the address value, the subnet mask value and the gateway value are displayed.

18. Storage media as in claim 17 wherein the software module is a plug-in module of the world wide web browser.

* * * * *

Network Working Group
Request for Comments: 2563
Category: Standards Track

R. Troll
@Home Network
May 1999

DHCP Option to Disable Stateless Auto-Configuration in IPv4 Clients

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

Operating Systems are now attempting to support ad-hoc networks of two or more systems, while keeping user configuration at a minimum. To accommodate this, in the absence of a central configuration mechanism (DHCP), some OS's are automatically choosing a link-local IP address which will allow them to communicate only with other hosts on the same link. This address will not allow the OS to communicate with anything beyond a router. However, some sites depend on the fact that a host with no DHCP response will have no IP address. This document describes a mechanism by which DHCP servers are able to tell clients that they do not have an IP address to offer, and that the client should not generate an IP address it's own.

1. Introduction

With computers becoming a larger part of everyday life, operating systems must be able to support a larger range of operating environments. One aspect of this support is the selection of an IP address. The Dynamic Host Configuration Protocol [DHCP] provides a superb method by which site administrators may supply IP addresses (and other network parameters) to network devices. However, some operating environments are not centrally maintained, and operating systems must now be able to handle this quickly and easily.

IPv6 accounts for this, and allows an IPv6 stack to assign itself a global address in the absence of any other mechanism for configuration [IPv6SAC]. However, Operating System designers can't wait for IPv6 support everywhere. They need to be able to assume

they will have IPv4 addresses, so that they may communicate with one another even in the smallest networks.

This document looks at three types of network nodes, and how IPv4 address auto-configuration may be disabled on a per-subnet (or even per-node) basis. The three types of network nodes are:

* A node for which the site administrator will hand out configuration information,

* A node on a network segment for which there is no site administrator, and

* A node on a network segment that has a central site administrator, and that administrator chooses not to hand out any configuration information to the node.

The difference between the second and third cases is the clients behavior.

In one case, the node may assign itself an IP address, and have full connectivity with other nodes on the local wire. In the last case, the node is not told what to do, and while it may assign itself a network address in the same way as case #2, this may not be what the central administrator wants.

The first scenario is handled by the current DHCP standard. However, the current DHCP specification [DHCP] says servers must silently ignore requests from hosts they do not know. Because of this, DHCP clients are unable to determine whether they are on a subnet with no administration, or with administration that is choosing not to hand out addresses.

This document describes a method by which DHCP clients will be able to determine whether or not the network is being centrally administrated, allowing it to intelligently determine whether or not it should assign itself a "link-local" address.

1.1. Conventions Used in the Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [KEYWORDS].

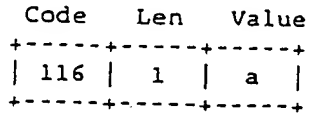
1.2. Terminology

DHCP client — A DHCP client is an Internet host using DHCP to obtain configuration parameters such as a network address.

DHCP server — A DHCP server is an Internet host that returns configuration parameters to DHCP clients.

2. The Auto-Configure Option

This option code is used to ask whether, and be notified if, auto-configuration should be disabled on the local subnet. The auto-configure option is an 8-bit number.



The code for this option is 116, and its length is 1.

This code, along with the IP address assignment, will allow a DHCP client to determine whether or not it should generate a link-local IP address.

2.1. Auto-Configure Values

The auto-configure option uses the following values:

DoNotAutoConfigure	0
AutoConfigure	1

When a server responds with the value "AutoConfigure", the client MAY generate a link-local IP address if appropriate. However, if the server responds with "DoNotAutoConfigure", the client MUST NOT generate a link-local IP address, possibly leaving it with no IP address.

2.2. DHCP Client Behavior

Clients that have auto-configuration capabilities MUST add the Auto-Configure option to the list of options included in its initial DHCPDISCOVER message. ([DHCP] Section 4.4.1) At this time, the option's value should be set to "AutoConfigure".

When a DHCPOFFER is received, it is handled as described in [DHCP], section 4.4.1, with one exception. If the 'yiaddr' field is 0x00000000, the Auto-Configure option must be consulted. If this

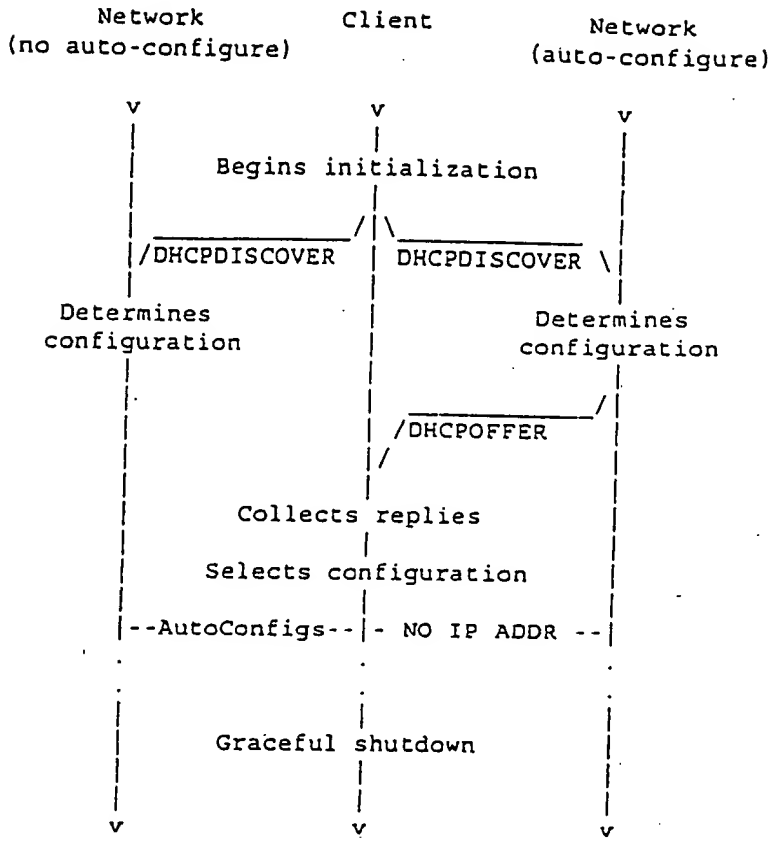
option is set to "AutoConfigure", then the DHCPOFFER MUST be ignored, and the DHCP client MAY generate a link-local IP address. However, if this option is set to "DoNotAutoConfigure", then the DHCPOFFER MUST be ignored, and the client MUST NOT generate a link-local IP address.

If a DHCP client receives any DHCPOFFER which contains a 'yiaddr' of 0x00000000, and the Auto-Configure flag says "DoNotAutoConfigure", in the absence of a DHCPOFFER with a valid 'yiaddr', the DHCP client MUST NOT generate a link-local IP address. The amount of time a DHCP client waits to collect any other DHCPOFFERs is implementation dependant.

DHCPOFFERs with a 'yiaddr' of 0x00000000 will only be sent by DHCP servers supporting the Auto-Configure option when the DHCPDISCOVER contained the Auto-Configure option. Since the DHCPDISCOVER will only contain the Auto-Configure option when a DHCP client knows how to handle it, there will be no inter-operability problems.

If the DHCP server does have an address to offer, the message states are the same as those described in [DHCP], section 3.

The following depicts the difference in responses for non-registered DHCP clients that support the "Auto-Configure" option on networks that have DHCP servers that support auto-configuration and networks with DHCP servers that do not.



2.3. DHCP Server Behavior

When a DHCP server receives a DHCPDISCOVER, it MUST be processed as described in [DHCP], section 4.3.1. However, if no address is chosen for the host, a few additional steps MUST be taken.

If the DHCPDISCOVER does not contain the Auto-Configure option, it is not answered.

If the DHCPDISCOVER contains the Auto-Configure option, and the site administrator has specified that Auto-Configuration should be disabled on the subnet the DHCPDISCOVER is originating from, or for the client originating the request, then a DHCPOFFER MUST be sent to the DHCP client. This offer MUST be for the address 0x00000000, and the Auto-Configure option MUST be set to "DoNotAutoConfigure".

If the site administrator allows auto-configuration on the originating subnet, the DHCPDISCOVER is not answered as before.

2.4. Mixed Environments

Environments containing a mixture of clients and servers that do and do not support the Auto-Configure option will not be a problem. Every DHCP transaction is between a Server and a Client, and the possible mixed scenarios between these two are listed below.

2.4.1. Client Supports, Server Does Not

If a DHCP client sends a request that contains the Auto-Configure tag, a DHCP server that does not know what this tag is will respond normally. According to [DHCP] Section 4.3.1, the server MUST NOT return a value for that parameter.

In this case, the server will either respond with a valid DHCPOFFER, or it will not respond at all. In both cases, a DHCP client that supports this option will never care what the state of the option is, and may auto-configure.

2.4.2. Servers Supports, Client Does Not

If the Auto-Configure option is not present in the DHCPDISCOVER, the server will do nothing about it. The client will auto-configure if it doesn't receive a response and believes that's what it should do.

This scenario SHOULD not occur, as any stacks that implement an auto-configuration mechanism MUST implement this option as well.

2.5. Interaction With Other DHCP Messages

As this option only affects the initial IP address selection, it does not apply to subsequent DHCP messages. If the DHCP client received a lease from a DHCP server, future DHCP messages (RENEW, INFORM, ACK, etc.) have no need to fall over into an auto- configuration state.

If the DHCP client's lease expires, the client falls back into the INIT state, and the initial DHCPDISCOVER is sent as before.

2.5.1. DHCPRELEASE Messages

DHCPRELEASEs occur exactly as described in [DHCP], section 4.4.6. When a DHCP client is done with a lease, it MAY notify the server that it is finished. For this to occur, the DHCP client already received a DHCP lease, and the state of Auto-Configuration on the local wire does not matter.

2.5.2. DHCPDECLINE Messages

A DHCPDECLINE is sent by the DHCP client when it determines the network address it is attempting to use is already in use. As a network address has been tested, it must have been offered by the DHCP Server, and the state of Auto-Configuration on the local wire does not matter.

2.5.3. DHCPINFORM Messages

DHCPINFORMs should be handled as described in [DHCP], section 4.4.3. No changes are necessary.

2.6. Message Option

If the DHCP server would like to tell a client why it is not allowed to auto-configure, it MAY add the Message option to the response. This option is defined in [DHCPOPT], Section 9.9.

If the DHCP client receives a response with the Message option set, it MUST provide this information to the administrator of the DHCP client. How this information is provided is implementation dependant.

3. Security Considerations

DHCP per se currently provides no authentication or security mechanisms. Potential exposures to attack are discussed in section 7 of the DHCP protocol specification [DHCP].

This mechanism does add one other potential attack. Malicious users on a subnet may respond to all DHCP requests with responses telling DHCP clients that they should NOT auto-configure on the local wire. On a network where Auto-Configuration is required, this will cause all DHCP clients to not choose an address.

4. Acknowledgments

This idea started at a joint Common Solutions Group / Microsoft meeting at Microsoft in May, 1998. The IP stacks in Win98 and NT5 assign themselves an IP address (in a specific subnet) in the absence of a responding DHCP server, and this is causing headaches for many sites that actually rely on machines not getting IP addresses when the DHCP servers do not know them.

Walter Wong proposed a solution that would allow the DHCP servers to tell clients not to do this. His initial solution would not work without slight modifications to DHCP itself. This document describes

those modifications.

5. IANA Considerations

The IANA has assigned option number 116 for this option.

6. References

[DHCP] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[DHCPOPT] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extension", RFC 2132, March 1997.

[IPv6SAC] Thomson, S. and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

7. Author's Address.

Ryan Troll
@Home Network
425 Broadway
Redwood City, CA 94063

Phone: (650) 556-6031
EMail: rtroll@corp.home.net

8. Full Copyright Statement

Copyright (C) The Internet Society (1999). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

Network Working Group
Request For Comments: 826

David C. Plummer
(DCP@MIT-MC)
November 1982

An Ethernet Address Resolution Protocol
-- or --
Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware

Abstract

The implementation of protocol P on a sending host S decides, through protocol P's routing mechanism, that it wants to transmit to a target host T located some place on a connected piece of 10Mbit Ethernet cable. To actually transmit the Ethernet packet a 48.bit Ethernet address must be generated. The addresses of hosts within protocol P are not always compatible with the corresponding Ethernet address (being different lengths or values). Presented here is a protocol that allows dynamic distribution of the information needed to build tables to translate an address A in protocol P's address space into a 48.bit Ethernet address.

Generalizations have been made which allow the protocol to be used for non-10Mbit Ethernet hardware. Some packet radio networks are examples of such hardware.

The protocol proposed here is the result of a great deal of discussion with several other people, most notably J. Noel Chiappa, Yogen Dalal, and James E. Kulp, and helpful comments from David Moon.

[The purpose of this RFC is to present a method of Converting Protocol Addresses (e.g., IP addresses) to Local Network Addresses (e.g., Ethernet addresses). This is a issue of general concern in the ARPA Internet community at this time. The method proposed here is presented for your consideration and comment. This is not the specification of a Internet Standard.]

Notes:

This protocol was originally designed for the DEC/Intel/Xerox 10Mbit Ethernet. It has been generalized to allow it to be used for other types of networks. Much of the discussion will be directed toward the 10Mbit Ethernet. Generalizations, where applicable, will follow the Ethernet-specific discussion.

DOD Internet Protocol will be referred to as Internet.

Numbers here are in the Ethernet standard, which is high byte first. This is the opposite of the byte addressing of machines such as PDP-11s and VAXes. Therefore, special care must be taken with the opcode field (ar$op) described below.

An agreed upon authority is needed to manage hardware name space values (see below). Until an official authority exists, requests should be submitted to

David C. Plummer
Symbolics, Inc.
243 Vassar Street
Cambridge, Massachusetts 02139

Alternatively, network mail can be sent to DCP@MIT-MC.

The Problem:

The world is a jungle in general, and the networking game contributes many animals. At nearly every layer of a network architecture there are several potential protocols that could be used. For example, at a high level, there is TELNET and SUPDUP for remote login. Somewhere below that there is a reliable byte stream protocol, which might be CHAOS protocol, DOD TCP, Xerox BSP or DECnet. Even closer to the hardware is the logical transport layer, which might be CHAOS, DOD Internet, Xerox PUP, or DECnet. The 10Mbit Ethernet allows all of these protocols (and more) to coexist on a single cable by means of a type field in the Ethernet packet header. However, the 10Mbit Ethernet requires 48.bit addresses on the physical cable, yet most protocol addresses are not 48.bits long, nor do they necessarily have any relationship to the 48.bit Ethernet address of the hardware. For example, CHAOS addresses are 16.bits, DOD Internet addresses are 32.bits, and Xerox PUP addresses are 8.bits. A protocol is needed to dynamically distribute the correspondences between a <protocol, address> pair and a 48.bit Ethernet address.

Motivation:

Use of the 10Mbit Ethernet is increasing as more manufacturers supply interfaces that conform to the specification published by DEC, Intel and Xerox. With this increasing availability, more and more software is being written for these interfaces. There are two alternatives: (1) Every implementor invents his/her own method to do some form of address resolution, or (2) every implementor uses a standard so that his/her code can be distributed to other systems without need for modification. This proposal attempts to set the standard.

Definitions:

Define the following for referring to the values put in the TYPE field of the Ethernet packet header:

ether_type$XEROX_PUP,
ether_type$DOD_INTERNET,
ether_type$CHAOS,

and a new one:

ether_type$ADDRESS_RESOLUTION.

Also define the following values (to be discussed later):

ares_op$REQUEST (= 1, high byte transmitted first) and
ares_op$REPLY (= 2),

and

ares_hrd$Ethernet (= 1).

Packet format:

To communicate mappings from <protocol, address> pairs to 48.bit Ethernet addresses, a packet format that embodies the Address Resolution protocol is needed. The format of the packet follows.

Ethernet transmission layer (not necessarily accessible to the user):

48.bit: Ethernet address of destination
48.bit: Ethernet address of sender
16.bit: Protocol type = ether_type$ADDRESS_RESOLUTION

Ethernet packet data:

16.bit:	(ar$hrd)	Hardware address space (e.g., Ethernet, Packet Radio Net.)
16.bit:	(ar$pro)	Protocol address space. For Ethernet hardware, this is from the set of type fields ether_typ$<protocol>.
8.bit:	(ar$hln)	byte length of each hardware address
8.bit:	(ar$pln)	byte length of each protocol address
16.bit:	(ar$op)	opcode (ares_op$REQUEST \| ares_op$REPLY)
nbytes:	(ar$sha)	Hardware address of sender of this packet, n from the ar$hln field.
mbytes:	(ar$spa)	Protocol address of sender of this packet, m from the ar$pln field.
nbytes:	(ar$tha)	Hardware address of target of this packet (if known).
mbytes:	(ar$tpa)	Protocol address of target.

Packet Generation:

As a packet is sent down through the network layers, routing determines the protocol address of the next hop for the packet and on which piece of hardware it expects to find the station with the immediate target protocol address. In the case of the 10Mbit Ethernet, address resolution is needed and some lower layer (probably the hardware driver) must consult the Address Resolution module (perhaps implemented in the Ethernet support module) to convert the <protocol type, target protocol address> pair to a 48.bit Ethernet address. The Address Resolution module tries to find this pair in a table. If it finds the pair, it gives the corresponding 48.bit Ethernet address back to the caller (hardware driver) which then transmits the packet. If it does not, it probably informs the caller that it is throwing the packet away (on the assumption the packet will be retransmitted by a higher network layer), and generates an Ethernet packet with a type field of ether_type$ADDRESS_RESOLUTION. The Address Resolution module then sets the ar$hrd field to ares_hrd$Ethernet, ar$pro to the protocol type that is being resolved, ar$hln to 6 (the number of bytes in a 48.bit Ethernet address), ar$pln to the length of an address in that protocol, ar$op to ares_op$REQUEST, ar$sha with the 48.bit ethernet address of itself, ar$spa with the protocol address of itself, and ar$tpa with the protocol address of the machine that is trying to be accessed. It does not set ar$tha to anything in particular, because it is this value that it is trying to determine. It could set ar$tha to the broadcast address for the hardware (all ones in the case of the 10Mbit Ethernet) if that makes it convenient for some aspect of the implementation. It then causes this packet to be broadcast to all stations on the Ethernet cable originally determined by the routing mechanism.

Packet Reception:

When an address resolution packet is received, the receiving Ethernet module gives the packet to the Address Resolution module which goes through an algorithm similar to the following. Negative conditionals indicate an end of processing and a discarding of the packet.

```
?Do I have the hardware type in ar$hrd?
Yes: (almost definitely)
  [optionally check the hardware length ar$hln]
  ?Do I speak the protocol in ar$pro?
  Yes:
    [optionally check the protocol length ar$pln]
    Merge_flag := false
    If the pair <protocol type, sender protocol address> is
        already in my translation table, update the sender
        hardware address field of the entry with the new
        information in the packet and set Merge_flag to true.
    ?Am I the target protocol address?
    Yes:
      If Merge_flag is false, add the triplet <protocol type,
          sender protocol address, sender hardware address> to
          the translation table.
      ?Is the opcode ares_op$REQUEST?  (NOW look at the opcode!!)
      Yes:
        Swap hardware and protocol fields, putting the local
            hardware and protocol addresses in the sender fields.
        Set the ar$op field to ares_op$REPLY
        Send the packet to the (new) target hardware address on
            the same hardware on which the request was received.
```

Notice that the <protocol type, sender protocol address, sender hardware address> triplet is merged into the table before the opcode is looked at. This is on the assumption that communcation is bidirectional; if A has some reason to talk to B, then B will probably have some reason to talk to A. Notice also that if an entry already exists for the <protocol type, sender protocol address> pair, then the new hardware address supersedes the old one. Related Issues gives some motivation for this.

Generalization: The ar$hrd and ar$hln fields allow this protocol and packet format to be used for non-10Mbit Ethernets. For the 10Mbit Ethernet <arhrd, arhln> takes on the value <1, 6>. For other hardware networks, the ar$pro field may no longer correspond to the Ethernet type field, but it should be associated with the protocol whose address resolution is being scught.

Why is it done this way??

Periodic broadcasting is definitely not desired. Imagine 100 workstations on a single Ethernet, each broadcasting address resolution information once per 10 minutes (as one possible set of parameters). This is one packet every 6 seconds. This is almost reasonable, but what use is it? The workstations aren't generally going to be talking to each other (and therefore have 100 useless entries in a table); they will be mainly talking to a mainframe, file server or bridge, but only to a small number of other workstations (for interactive conversations, for example). The protocol described in this paper distributes information as it is needed, and only once (probably) per boot of a machine.

This format does not allow for more than one resolution to be done in the same packet. This is for simplicity. If things were multiplexed the packet format would be considerably harder to digest, and much of the information could be gratuitous. Think of a bridge that talks four protocols telling a workstation all four protocol addresses, three of which the workstation will probably never use.

This format allows the packet buffer to be reused if a reply is generated; a reply has the same length as a request, and several of the fields are the same.

The value of the hardware field (ar$hrd) is taken from a list for this purpose. Currently the only defined value is for the 10Mbit Ethernet (ares_hrd$Ethernet = 1). There has been talk of using this protocol for Packet Radio Networks as well, and this will require another value as will other future hardware mediums that wish to use this protocol.

For the 10Mbit Ethernet, the value in the protocol field (ar$pro) is taken from the set ether_type$. This is a natural reuse of the assigned protocol types. Combining this with the opcode (ar$op) would effectively halve the number of protocols that can be resolved under this protocol and would make a monitor/debugger more complex (see Network Monitoring and Debugging below). It is hoped that we will never see 32768 protocols, but Murphy made some laws which don't allow us to make this assumption.

In theory, the length fields (ar$hln and ar$pln) are redundant, since the length of a protocol address should be determined by the hardware type (found in ar$hrd) and the protocol type (found in ar$pro). It is included for optional consistency checking, and for network monitoring and debugging (see below).

The opcode is to determine if this is a request (which may cause a reply) or a reply to a previous request. 16 bits for this is overkill, but a flag (field) is needed.

The sender hardware address and sender protocol address are absolutely necessary. It is these fields that get put in a translation table.

The target protocol address is necessary in the request form of the packet so that a machine can determine whether or not to enter the sender information in a table or to send a reply. It is not necessarily needed in the reply form if one assumes a reply is only provoked by a request. It is included for completeness, network monitoring, and to simplify the suggested processing algorithm described above (which does not look at the opcode until AFTER putting the sender information in a table).

The target hardware address is included for completeness and network monitoring. It has no meaning in the request form, since it is this number that the machine is requesting. Its meaning in the reply form is the address of the machine making the request. In some implementations (which do not get to look at the 14.byte ethernet header, for example) this may save some register shuffling or stack space by sending this field to the hardware driver as the hardware destination address of the packet.

There are no padding bytes between addresses. The packet data should be viewed as a byte stream in which only 3 byte pairs are defined to be words (arhrd, arpro and ar$op) which are sent most significant byte first (Ethernet/PDP-10 byte style).

Network monitoring and debugging:

The above Address Resolution protocol allows a machine to gain knowledge about the higher level protocol activity (e.g., CHAOS, Internet, PUP, DECnet) on an Ethernet cable. It can determine which Ethernet protocol type fields are in use (by value) and the protocol addresses within each protocol type. In fact, it is not necessary for the monitor to speak any of the higher level protocols involved. It goes something like this:

When a monitor receives an Address Resolution packet, it always enters the <protocol type, sender protocol address, sender hardware address> in a table. It can determine the length of the hardware and protocol address from the ar$hln and ar$pln fields of the packet. If the opcode is a REPLY the monitor can then throw the packet away. If the opcode is a REQUEST and the target protocol address matches the protocol address of the monitor, the monitor sends a REPLY as it normally would. The monitor will only get one mapping this way, since the REPLY to the REQUEST will be sent directly to the requesting host. The monitor could try sending its own REQUEST, but this could get two monitors into a REQUEST sending loop, and care must be taken.

Because the protocol and opcode are not combined into one field, the monitor does not need to know which request opcode is associated with which reply opcode for the same higher level protocol. The length fields should also give enough information to enable it to "parse" a protocol addresses, although it has no knowledge of what the protocol addresses mean.

A working implementation of the Address Resolution protocol can also be used to debug a non-working implementation. Presumably a hardware driver will successfully broadcast a packet with Ethernet type field of ether_type$ADDRESS_RESOLUTION. The format of the packet may not be totally correct, because initial implementations may have bugs, and table management may be slightly tricky. Because requests are broadcast a monitor will receive the packet and can display it for debugging if desired.

An Example:

Let there exist machines X and Y that are on the same 10Mbit Ethernet cable. They have Ethernet address EA(X) and EA(Y) and DOD Internet addresses IPA(X) and IPA(Y) . Let the Ethernet type of Internet be ET(IP). Machine X has just been started, and sooner or later wants to send an Internet packet to machine Y on the same cable. X knows that it wants to send to IPA(Y) and tells the hardware driver (here an Ethernet driver) IPA(Y). The driver consults the Address Resolution module to convert <ET(IP), IPA(Y)> into a 48.bit Ethernet address, but because X was just started, it does not have this information. It throws the Internet packet away and instead creates an ADDRESS RESOLUTION packet with

```
(ar$hrd) = ares_hrd$Ethernet
(ar$pro) = ET(IP)
(ar$hln) = length(EA(X))
(ar$pln) = length(IPA(X))
(ar$op)  = ares_op$REQUEST
(ar$sha) = EA(X)
(ar$spa) = IPA(X)
(ar$tha) = don't care
(ar$tpa) = IPA(Y)
```

and broadcasts this packet to everybody on the cable.

Machine Y gets this packet, and determines that it understands the hardware type (Ethernet), that it speaks the indicated protocol (Internet) and that the packet is for it ((ar$tpa)=IPA(Y)). It enters (probably replacing any existing entry) the information that <ET(IP), IPA(X)> maps to EA(X). It then notices that it is a request, so it swaps fields, putting EA(Y) in the new sender Ethernet address field (ar$sha), sets the opcode to reply, and sends the packet directly (not broadcast) to EA(X). At this point Y knows how to send to X, but X still doesn't know how to send to Y.

Machine X gets the reply packet from Y, forms the map from <ET(IP), IPA(Y)> to EA(Y), notices the packet is a reply and throws it away. The next time X's Internet module tries to send a packet to Y on the Ethernet, the translation will succeed, and the packet will (hopefully) arrive. If Y's Internet module then wants to talk to X, this will also succeed since Y has remembered the information from X's request for Address Resolution.

Related issue:

It may be desirable to have table aging and/or timeouts. The implementation of these is outside the scope of this protocol. Here is a more detailed description (thanks to MOON@SCRC@MIT-MC).

If a host moves, any connections initiated by that host will work, assuming its own address resolution table is cleared when it moves. However, connections initiated to it by other hosts will have no particular reason to know to discard their old address. However, 48.bit Ethernet addresses are supposed to be unique and fixed for all time, so they shouldn't change. A host could "move" if a host name (and address in some other protocol) were reassigned to a different physical piece of hardware. Also, as we know from experience, there is always the danger of incorrect routing information accidentally getting transmitted through hardware or software error; it should not be allowed to persist forever. Perhaps failure to initiate a connection should inform the Address Resolution module to delete the information on the basis that the host is not reachable, possibly because it is down or the old translation is no longer valid. Or perhaps receiving of a packet from a host should reset a timeout in the address resolution entry used for transmitting packets to that host; if no packets are received from a host for a suitable length of time, the address resolution entry is forgotten. This may cause extra overhead to scan the table for each incoming packet. Perhaps a hash or index can make this faster.

The suggested algorithm for receiving address resolution packets tries to lessen the time it takes for recovery if a host does move. Recall that if the <protocol type, sender protocol address> is already in the translation table, then the sender hardware address supersedes the existing entry. Therefore, on a perfect Ethernet where a broadcast REQUEST reaches all stations on the cable, each station will be get the new hardware address.

Another alternative is to have a daemon perform the timeouts. After a suitable time, the daemon considers removing an entry. It first sends (with a small number of retransmissions if needed) an address resolution packet with opcode REQUEST directly to the Ethernet address in the table. If a REPLY is not seen in a short amount of time, the entry is deleted. The request is sent directly so as not to bother every station on the Ethernet. Just forgetting entries will likely cause useful information to be forgotten, which must be regained.

Since hosts don't transmit information about anyone other than themselves, rebooting a host will cause its address mapping table to be up to date. Bad information can't persist forever by being passed around from machine to machine; the only bad information that can exist is in a machine that doesn't know that some other machine has changed its 48.bit Ethernet address. Perhaps manually resetting (or clearing) the address mapping table will suffice.

This issue clearly needs more thought if it is believed to be important. It is caused by any address resolution-like protocol.